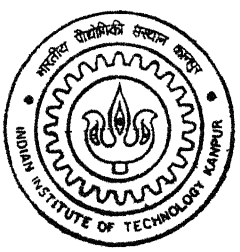
Analysis of Crossover Operators for Real-coded Genetic Algorithms

by

Amit Kumar Gupta

TH
ME/2001/M
G959a



DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January, 2001

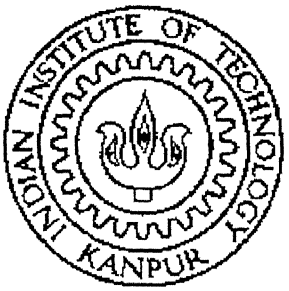
Analysis of Crossover Operators for Real-coded Genetic Algorithms

A thesis submitted
in partial fulfillment of the requirements
for the degree of

Master of Technology

by

Amit Kumar Gupta



to the

Department of Mechanical Engineering
Indian Institute of Technology
Kanpur-208016, India
January, 2001

A133712

Acknowledgment

I would like to express my sincere gratitude to my thesis guide and respected sir Prof. Kalyanmoy Deb for his constant help and motivation to accomplish the present work. I also would like to thank him for clarifying the doubts and problems in my thesis which helped in the completion of this work. I also feel highly obliged to him for giving me time at the late night hours for discussions and problems in my thesis.

I acknowledge the company of my lab members, Tushar, Saket, Amrit and others who helped me a lot. I also could not forget Sarvesh, Mukesh, Ruchin, Sudhir and all my batch-mates who made my stay at IIT Kanpur a memorable one. I also thank all the people particularly Ruchin who have directly or indirectly helped in the completion of the thesis.

Amit Kumar Gupta

Abstract

Genetic algorithms are the optimization procedures which mimic the working of nature in solving problems In their earlier form, the problem variables are represented in binary notation This GA is called binary-coded genetic algorithm (BCGA) However, there are certain problems associated with this form of GAs Other form of coding such as real number representation have been suggested for use in GA This representation seems to be natural for optimizing problems having variables in continuous domain

Real-coded genetic algorithms (RCGAs) have been found to be very efficient in optimizing real world problems These use real number representation for the problem variables The RCGAs have several advantages over the traditional binary-coded genetic algorithms (BCGA) The RCGAs differ from the BCGAs in the coding of problem variables, the crossover operator and the mutation operator

The crossover operator is believed to be the main search operator in the working of a genetic algorithm since it is primarily responsible for creating the children population in the search space The performance of a real-coded genetic algorithm largely depends on how good the crossover operator is in exploring and exploiting the search space A number of crossover operators for RCGAs are available in the literature Since the crossover operator is so important, therefore it essential to find out the desirable properties of a crossover operator that it should have for the algorithm to perform efficiently

In the present work first an description of different evolutionary algorithms using real number representation has been given and then after a number of crossover operators available for RCGA have been analyzed taking the postulates suggested by Beyer and Deb (1999) as the base The postulates are on the mean preservation and variance evolution characteristics of the evolutionary algorithms Based on the postulates, the crossover operators have been analyzed for flat fitness landscape The variance evolution equations for these operators have been obtained The values of the characteristic parameter values associated with different crossover operators have been found for which the various RCGAs implementing different crossover operators will perform similarly in flat fitness landscape These different algorithms have been tested for their similar performance on other landscape taking values for flat fitness landscape Later the crossover operators have been classified based on their mean preservation and variance evolution characteristics and guidelines for the design of an efficient crossover operator has been suggested

Contents

1 Introduction 2

1 1 Literature Survey 2

1 1 1 Binary Coded Genetic Algorithms 3

1 1 2 Real Coded Genetic Algorithms 5

1 2 Objective 6

1 3 Organization of the Thesis 6

2 Real Parameter Evolutionary Algorithms 8

2 1 Real Coded Genetic Algorithms (*RCGA*) 8

2 1 1 Schemata Concept as related to Real-coded Genetic Algorithms 9

2 1 2 Crossover Operators 13

2 1 3 Two Parent Recombination 14

2 1 4 Multi Parent Recombination 25

2 1 5 Mutation Operators 28

2 2 Variants of Real-coded Genetic Algorithm 31

2 2 1 Genetic Adaptive Search (GeneAS), (Deb et al , 1996) 31

2 2 2 BUGS (Iba et al , 1993) 32

2 3 Evolution Strategies 34

2 3 1 Representation 34

2 3 2 Mutation 35

2 3 3 Recombination 36

2 3 4 Selection 37

2 3 5 Invariance Property of Evolution Strategies 37

2 4 Differential Evolution 38

2 4 1 Initialization 39

2 4 2 Mutation 39

2 4 3 Recombination 39

2 4 4 Selection 39
2 5 Family Competition Evolutionary Algorithm (FCEA) 40
2 5 1 Representation 41
2 5 2 Selection Procedures 41
2 5 3 Recombination 42
2 5 4 Mutation 43

3 Behaviour of Crossover Operators for Real-Coded GA 45
3 1 Need 45
3 1 1 Postulates on Population Mean 45
3 1 2 Postulates on Population Variance 46
3 2 Analysis of Crossover Operators for Real-coded GAs in Flat Fitness Landscapes 47
3 2 1 Determination of Variance Evolution Equation for Different Crossover Operator 49
3 2 2 Comparison with Simulation Results 62
3 3 Classification of Crossover Operators 71
3 3 1 Discussion 71

4 Extensions to Other Landscapes 73
4 1 Sphere Model 73
4 2 Ridge Model 76
4 3 Discussion 78

5 Conclusion and Scope for Future Work 81
5 1 Conclusion 81
5 2 Scope for Future Work 84

Appendix A 86

Bibliography 88

List of Figures

1 1 A simple unimodal function . 4

2 1 A mean slice, its basins, its features and its virtual characters (Taken from Goldberg, 1990) 11

2 2 The cross-hatched regions are likely to be processed by the algorithms (Taken from Goldberg, 1990) 12

2 3 The children points are generated along the diagonal formed by parents 14

2 4 Simple Crossover 15

2 5 Arithmetical Crossover 16

2 6 BLX-α The children points are randomly uniformly picked from the rectangle formed by the two parents 17

2 7 Linear Crossover 17

2 8 Extended Line Crossover Generates points on the extended diagonal formed by parents 18

2 9 Wright's heuristic operator Parent1 is better in fitness 19

2 10 Linear BGA crossover The child is randomly picked from the thick portion near first parent 19

2 11 Unfair average crossover 20

2 12 Fuzzy Connective Based Crossover Operators 21

2 13 Fuzzy Recombination Operator, d = 0 5 22

2 14 Box BLX-α-β 23

2 15 Directional BLX-α-β, The children are uniformly randomly selected from the trimmed rectangle 23

2 16 SBX Crossover Operator Probability Distribution for the generation of Children 24

2 17 UNDX Operator 26

2 18 The SPX Operator The children points are chosen from within the extended simplex 28

2 19 Probability distribution for creating a mutated value for continuous variables 30

2 20 The family competition evolutionary algorithm and procedure used to do the mutation, recombination and selection 41

3 1 Evolution of the square root of the expected population variance for the SBX operator in the flat fitness landscape Population size=10 with $\eta = 2$ (upper line, displayed by +, simulations displayed by $\diamond$) and $\eta = 5$ (lower, dotted line simulations displayed by $\times$) The population has been randomly initialized with variance 1 The simulation results have been obtained by averaging over 3000 independent runs 63

3 2 Evolution of the square root of the expected population variance for the SBX operator in the flat fitness landscape Population size=300 with $\eta = 2$ (upper line, displayed by +, simulations displayed by $\diamond$) and $\eta = 10$ (lower, dotted line, simulations displayed by $\times$) The population has been randomly initialized with variance 6 The simulation results have been obtained by averaging over 100 independent runs 63

3 3 Evolution of the square root of the expected population variance for the BLX operator in the flat fitness landscape BLX-0 0 for population size 60 (upper line displayed by +, simulations displayed with a $\diamond$) and 4 (lower dotted line, simulations displayed with a $\times$) The simulation started with random initialization with variance 1 The results have been obtained by averaging over 2000 (popsize 4) and over 100 (popsize 60) independent runs, respectively 64

3 4 Evolution of the square root of the expected population variance for the BLX operator in the flat fitness landscape BLX-0 5 for population size 60 (upper line displayed by + simulations displayed with a $\diamond$) and 4 (lower dotted line, simulations displayed with a $\times$) The simulation started with random initialization with variance 1 The results have been obtained by averaging over 2000 (popsize 4) and over 100 (popsize 60) independent runs, respectively 64

3 5 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 5$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs 65

3 6 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 7$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs 65

3 7 Evolution of the square root of the expected population variance for the UNDX crossover operator in the flat fitness landscape with $\sigma_\xi = 0\ 75$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 6 with $\sigma_\xi = 0\ 55$ (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=6) and 200 (popsize=60) independent runs 66

3 8 Evolution of the square root of the expected population variance for UNDX operator in the flat fitness landscape with $\sigma_\xi = 0\ 50$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 10 with $\sigma_\xi = 0\ 35$ (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=10) and 200 (popsize=60) independent runs 66

3 9 Evolution of the square root of the expected population variance for the SPX operator with two parents with $\epsilon = 1\ 5$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=6 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=6) and 300 (popsize=60) independent runs 67

3 10 Evolution of the square root of the expected population variance for the SPX operator with two parents with $\epsilon = 2\ 0$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=6 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=6) and 300 (popsize=60) independent runs 67

3 11 Evolution of the square root of the expected population variance for the SPX operator with three parents with $\epsilon = 2\ 1$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=10) and 200 (popsize=60) independent runs 68

3 12 Evolution of the square root of the expected population variance for the SPX operator with three parents with $\epsilon = 2.25$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=5 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs 68

3 13 Evolution of the square root of the expected population variance for the SPX operator with five parents with $\epsilon = 2.5$ for population size=120 (upper line shown with +, simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs 69

3 14 Evolution of the square root of the expected population variance for the SPX operator with five parents with $\epsilon = 2.75$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs 69

3 15 The performance of various RCGAs implementing different operators in flat fitness landscape The dotted line is the theoretical line From top the operators are SPX, BLX SBX, FR and UNDX 70

4 1 Performance of several real-coded GAs with different crossover operators on sphere function 74

4 2 Performance of UNDX operator on sphere model after the σ_ξ value is reduced 75

4 3 Performance of SPX operator on sphere model after the ϵ value is reduced 75

4 4 The performance of UNDX and SPX operators on sphere model with variable by variable crossover 76

4 5 Ridge function 76

4 6 The performance of various RGAs with different operators on parabolic ridge function 77

4 7 RCGAs using SBX and BLX operators performing similar to FR operator on ridge model after altering the associated parameter value 78

4 8 Performance of UNDX operator on ridge function after increasing the value of σ_ξ 79

4 9 Performance of SPX operator on ridge function after increasing the value of ϵ 79

List of Tables

2 1 Fuzzy Connective Based Crossover Operators 21

3 1 Characteristic parameter values for different crossover operators for similar bahaviour in flat fitness landscape 70

3 2 Crossover Operators for Real-coded Genetic Algorithms 71

3 3 Logical fuzzy connective used in FCB operators 72

Chapter 1

Introduction

Real-Coded Genetic Algorithms($RCGA$) are the search algorithms based on natural evolution These have been found to be robust search algorithms to solve complicated problems The $RCGA$ works with a population of the candidate solutions of the problem, applying several operators on this population These operators are

- Selection
- Crossover
- Mutation

The crossover operator is believed to be the main search operator in the working of a genetic algorithm since it is responsible for creating new children from the population given by the selection operator The performance of a genetic algorithm largely depends on how good the crossover operator is in exploring new regions in the search space by exploiting the regions as given by the selection operator So it is essential to have a good crossover operator to explore and exploit the search space so as to give faster convergence to the optima avoiding premature convergence Mutation operator is also used in GA but it is used only as a backhand operator with small probability to perturb the value of the parent by a little amount

There are a number of crossover operators that have been proposed by many researchers for real-coded genetic algorithms Each one of these operators has been claimed to be performing better on certain test functions in comparison to some previous operator Since the crossover operator plays an important role in the search procedure so it is necessary to find out properties of a crossover operator with which the algorithm could perform efficiently in optimizing the problems

1.1 Literature Survey

The genetic algorithms(GA) are the search algorithms based on the natural evolution The GAs work on a population of the candidate solutions for the problem to be optimized The main steps in

the working of a GA are the selection, the crossover and the mutation as have been pointed above

1 1 1 Binary Coded Genetic Algorithms

The earlier form of the *GAs* is the binary-coded genetic algorithms (*BCGA*) In BCGA the problem vector is coded in the binary bits i e (0,1) The variables are transformed in to the binary presentation and concatenated so as to form one big string which represents a vector This concatenated string is called an "*individual*" The GA start with a population of these randomly generated individuals The variables are transformed back to the real space by using a linear mapping as below

$$x_i = x_i^l + \frac{(x_i^u - x_i^l)}{2^{l_i} - 1} DV$$

Here i varies from 1 to n (no of variables) x_i^l and x_i^u are the lower and upper bounds on the variable x_i respectively, l_i is the string length for the i^{th} variable and DV is the decoded value of the binary representation for this variable With the binary coding, the accuracy that can be obtained in the solutions is given by the fraction term of the above equation Thus *BCGA* works by discretising the search space

The selection operator in the GA works on the principle of *"Survival of Fittest"* The fitness here is the value of objective function that is to be optimized in maximization problem In minimization problem the fitness is usually defined as the function $\frac{1}{1+obj}$ where obj is the objective function value at the particular point The selection operator chooses the fitter solutions to form the mating population The different selection techniques used are,

- Roulette Wheel Selection
- Stochastic Remainder Selection
- Tournament Selection

On the mating pool of the individuals as given by the selection operator, the crossover and mutation operators work to form the population for the next generation In the crossover two parents are randomly selected from the mating pool and a cross site is randomly generated, the two children solution are obtained by swapping the portions of these parents that are on the right of the cross site This operator is known as the "single point crossover" There are other crossover operators for *BCGA* also available as n-point crossover and uniform crossover The mutation operator works by randomly inverting the bit of an individual with a certain small probability This forms the population for the next generation Since the *BCGAs* work with a binary coding of the variables, these discretize the search space in solving the optimization problems having continuous search space This discretisation results in problems in the *BCGA* The problems associated with the *BCGA* are described next,

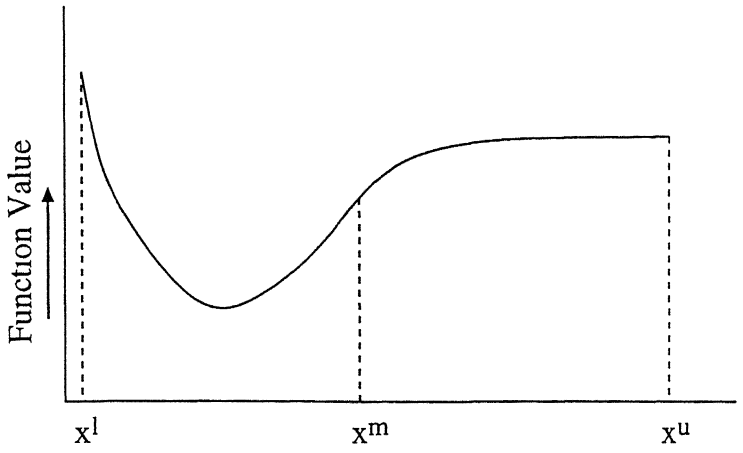


Figure 1 1 A simple unimodal function

1 Any arbitrary precision in the result can not be achieved instead the precision in the result depends on the length of the string used to represent the variable

2 The *BCGA* needs the knowledge of the bounds on the problem parameters to do the linear mapping from the binary to real search space The knowledge of the bounds on the problem parameters generally is not available in the real world problems

3 Inherent Hamming-cliff problem associated with the binary coding

4 All Holland schemata are not important

The first point is because the *BCGA* discretizes the search space for creating the solutions This limit the precision that can be achieved in the solution For more precision we need to use larger string length with results in large computation time

Since we have to do the linear mapping of the binary representation to the real space to get the values of parameters, we need to know the bounds on the problem variables This is not likely in the real-world problems

With the binary representation, to obtain a neighouring solution we may have to flip several bits This may cause the GA to take longer computation time and also the GA may get stuck

The fourth point may be explained with the help of Figure 1 1 Considering it to be a minimization problems, from the figure it is clear that the left portion of the search space between lower and upper bound contains the global optima The the bit strings having zero at their left are more important since these are the representative of the left half of the search space

Thus the schemata which represent the left half are more important than those which represent the right space So the processing of the schemata having 1 at the leftmost position is not needed here

Since the binary coded GAs search only within the given lower and upper bounds, these are not suitable for the problems where the range in which the optima lies is not known apriori since in this case it is necessary to generate the children points outside the initial bounds These are also not suitable for the problems in which the optima is time dependent

1 1.2 Real Coded Genetic Algorithms

The real-coded genetic algorithm($RCGA$) differs in many ways from the binary-coded GA as follows,

- Coding of variables
- Crossover
- Mutation

These differences are described in the following

Coding of Variables

In $RCGA$, the variables are directly coded into real numbers So a concatenated string i e an individual which is a collection of binary bits 0 and 1 in case of $BCGA$, is a collection of real numbers in $RCGA$ Thus the problem of achieving arbitrary precision with the $BCGA$ is eliminated in $RCGA$ since the floating point representation is directly used The real number representation of the problem parameters also eliminates the problem of *Hamming-cliff* (the need of alteration of many bits in order to get a neighboring solution in real space) with $BCGA$

Crossover

The crossover operation in the real coded genetic algorithm is different from that in the $BCGA$ since we are using real numbers to represent the variables, the operators which are used in $BCGA$ would not be effective in the $RCGA$ Here the crossover is done using some distribution of the population and the children points are created using this distribution One such crossover operator is *flat crossover* in which one offspring is created as,

$$y_i = (1 - \alpha)x_i^1 + \alpha x_i^2$$

Here y_i is the child, x_i^1 and x_i^2 are the parents α is a uniformly generated random number in the range (0,1) Here this crossover operator generates a child uniformly and randomly from the span bounded by the two parents Other crossover operators are described in Chapter 2

Mutation

Like crossover, the mutation operation also uses a probability distribution in generating a point from the parent It perturbes the value of a variable by a little amount to some neighouring value One mutation operator which has been proposed for the *RCGA* is the "Non-uniform Mutation" by Michalewicz,1992 This works as follows,

If the operator is being applied at generation g and g_{max} is the maximum number of generations then

$$y'_i = \begin{cases} y_i + \triangle(g, b_i - y_i) & \text{if } \tau = 0 \\ y_i - \triangle(g, y_i - a_i) & \text{if } \tau = 1 \end{cases}$$

Here a_i and b_i are the lower and upper bounds on the i^{th} variable τ is a random number which can have a value 0 or 1 and the operator $\triangle$ is defined as,

$$\triangle(g, y) = y\left(1 - r^{\left(1 - \frac{g}{g_{max}}\right)^b}\right)$$

where $r \in (0, 1)$ and b is a parameter chosen by the user Other mutations operators have been described in Chapter 2

1.2 Objective

The present work aims at determination of the desirable properties that a crossover operator should posses to perform efficiently There are a number of crossover operators for the real-coded genetic algorithms available in the literature, proposed by different researchers

In the present work, the mean and variance of different crossover operators has been explored for the flat fitness landscape case on the basis of postulates proposed by Deb and Beyer, 1999 The different crossover operators are then checked whether they follow the postulates The analysis includes checking whether a crossover operator preserves the mean of the parental population and the determination of the variance evolution equation for different crossover operators The values of the characteristic parameter associated with different operators is also presented for which the different operators will perform similarly in the flat fitness landscape Thenafter the GA runs implementing different crossover opertors have been carried out on the landsapes other than flat fitness landscape to check the significance of the values obtained for flat fitness landsape Finally the operators are classified according to the properties they posses and suggestions for the design of crossover operators is presented

1.3 Organization of the Thesis

The thesis is organized as follows

In Chapter 2, different evolutionary algorithms which use the real number representation of the problem parameters have been described These algorithms include "Real-coded Genetic Algorithms($RCGA$) Evolution Strategies(ES)", "Differential Evolution(DE)" and "Family Competition Evolutionary Algorithm($FCEA$)"

In Chapter 3, the analysis of different crossover operator for the real coded genetic algorithms is performed and the theoretical predictions are compared with the simulations

Chapter 4 includes the results that have been obtained by running simulations on sphere and ridge model, for different RCGAs having different crossover operators

Chapter 5 includes the conclusion and discussion on the behaviour of different $RCGAs$ and the suggestions for the future work

Chapter 2

Real Parameter Evolutionary Algorithms

In this chapter the *evolutionary algorithms* using real number representation are discussed First the real-coded genetic algorithm (*RCGA*) is discussed and then after the evolution strategies (*ES*), differential evolution (*DE*) & family competition evolutionary algorithm (*FCEA*) are presented

2.1 Real Coded Genetic Algorithms (*RCGA*)

As discussed in Chapter1, the *RCGA* works with a population of randomly generated individuals of the problem vector applying operators like selection, crossover and mutation The performance of the *RCGA* is largely dependent on the crossover operator since it is responsible for creating the solutions using the parental population Mutation operator is also used but it is meant only for perturbing a particular solution locally by a little amount Crossover operator is used for finding the global solution The problems with the earlier version of GA i e *BCGA* have already been discussed in Chapter 1 *RCGA* eliminates these problems and has following advantages,

1 With the use of real numbers directly to represent the variables, it is possible to use large domains in the search specially in the problems in which the location of the optimum is not known which is generally the case In *BCGA* if larger domain are used it is necessary to use larger string length which results in higher computation cost and time

2 The inherent *Hamming-cliff* problem is completely removed in the *RCGA* because of using real numbers directly

3 The use real numbers in representing the problem parameter facilitates the achievement of any desired accuracy in the solution

In the following the different concepts and operators used in real-coded genetic algorithms are described

2 1 1 Schemata Concept as related to Real-coded Genetic Algorithms

The main theoretical advantage of the binary coded genetic algorithms is that the processing of *BCGA* is well defined on the basis of schema theorem The schema as defined for *BCGA* is a similarity template that describes a subset of the strings with similar features at certain positions The schema is a string defined over the length of the individual with mainly three values a zero, an one and an asterisk(⋆) The ⋆ is a wild card or don't care symbol i e it represents either a zero or an one In binary GA, it has been shown theoretically that the above average schemata get more copies in the coming generations For an example an schema for an string of length 7 may be ⋆11⋆010 It is an representative of following four solutions {(0110010),(1110010),(0111010),(1111010)} The order of a schema is defined as the number of don't care symbols available in the schema For the above schema the order is 2 The other term which has been defined for the schema is its defining length It is difference of positions of the two extreme don't care symbols

The concept of schema for real coded GAs have also been proposed by many researchers The different terms proposed for analysing the *RCGA* are presented below

Wright's Connected Schema

In (Wright, 1991), Wright studied the meaning of the classic binary-coded schemata for continuous functions in terms of real parameters Wright defined the meaning of binary single point crossover as the generation of child having more significant part of one parent and less significant part of the other Thus he argued that on the real space the child can be considered as a perturbation of the one of the parents The maximum size of the perturbation for parameter i is $R_i 2^{-k}$ Here R_i is the range of the parameter and k is the cross site chosen to do the crossover The perturbation obtained correspond to the difference of the less significant bits i e bits on the right of the cross site

He defined the schemata which have contiguous *'s on the right end of string as connected intervals of the real numbers The schema whose *'s are not all contiguous at the right end corresponds to a disconnected union of intervals He defined these schemata which represents the connected intervals of the real parameters as "connected schemata"

He defined the schemata for the real genes by restricting some or all of the parameters to subintervals of then possible ranges If the variable i has lower and upper bounds as a_i and b_i the parameter space can be represented as,

$$p = \prod_{i=1}^{i=n} [a_i, b_i]$$

and a schema s is specified by choosing real bluer α_i and β_i such that $a_i \leq \alpha_i \leq \beta_i \leq b_i$ To each schema s a template is associated as below

If a parameter is unrestricted i e $\alpha_i = a_i$ and $\beta_i = b_i$, a * is associated with this schema If a parameter is restricted i e $\alpha_i > a_i$ and $\beta_i < b_i$, I_i denotes its interval So an schema is given as

$$\prod_{i=1}^{i=n} [\alpha_i, \beta_i]$$

For example if the parameter space is $[-1, 1] \times [-1, 1] \times [-1, 1]$ and is the schema is $[-1, 1] \times [0, 1] \times [-1\ 0]$ the corresponding n-tuple would be $*I_2 I_3$

Virtual Alphabets and Blocking

To explain the working of real coded genetic algorithms Goldberg (Goldberg, 1990) proposed the theory of *virtual characters* and *virtual alphabets* He tried to make it compatible with the schema theorem for binary coded GA so as to predict the behaviour of the *RCGA*

Goldberg postulates that in the working of a genetic algorithm, *selection* dominates the GA search by cutting the search space in to domains where the points having above average fitness reside So in the early phases of search, the search space gets reduced to certain regions where the fitness is above average It has been already shown (in Goldberg and Deb 1990) that the number of generations needed for the above average points to take over the populations is $g = \log\log n_p$ Here n_p is the population size and g represents the number of generations So even if the population size is 10^9, the number of generations for the selection to dominate is merely 5

He explained the action of *selection* in terms of *slices* of the fitness function The different terms are stated below,

Slices The *mean slice* represents the average value of the fitness function averaged over the intervals of variables Mathematically,

$$\overline{\phi}(x_1) = \frac{1}{b-a} \int_a^b \phi(x_1, x_2)\, dx_2$$

The *mean slice* can be obtained for different values of x_1 from the above function Similarly the *global slice* can be defined by fixing the value of x_2 at global optimum i e $\phi^*(x_1) = \phi(x_1, x_2^*)$ The *max slice* may be obtained by varying x_1 and choosing the best ϕ value at each point ranging over all possible x_2 values

Considering an appropriate probability density function $f(x)$, $x \in \Omega$, the marginal density function $f_I(x_I)$ for the subset of n real variables specified by the index set I may be calculated as

$$f_I(x_I) = \int_{\Omega_{I'}} f(x) d\Omega_{I'}$$

where $I' = \{1, \quad n\} - I$ The *mean slice* $\overline{\phi}_I$ or the expected value of ϕ with respect to f and the free variables x_i, $i \in I$ may be calculated as follows

$$\overline{\phi}_I(x_I) = \int_{\Omega'_I} f_{I'}(x_{I'})\phi(x)d\Omega_{I'}$$

where $d\Omega_{I'} = \prod_{j\in I'} dx_j$ and $\Omega_{I'}$ is the indicated $|I'|$-dimensional subspace of Ω spanned by fixed variables A point slice is defined as the function obtained by setting $x_j = x_j$ for point $x = a$ with respect to the index set I and allowing other variables vary freely

Virtual Characters and Alphabets With the definitions of *slices* and the functions developed above a rigorous statement regarding the above average intervals was developed Considering only the *mean slices* of ϕ over the singleton index sets, the one-dimensional *mean slices* $\overline{\phi}_i$, a *virtual character* may be defined as a non-empty intersection of a basin and a feature on ϕ_i, and a *virtual alphabet* is simply the collection of virtual characters along a given direction The figure 2 1 shows these concepts In the figure the selection set is divided into two features and four basins The first feature (F_1) contains portions of two different basins (B_1 and B_2) and is divided into two virtual characters (C_1 and C_2) The second feature (F_2) contains a portion of only one basin and as a result has only a single character (C_3)

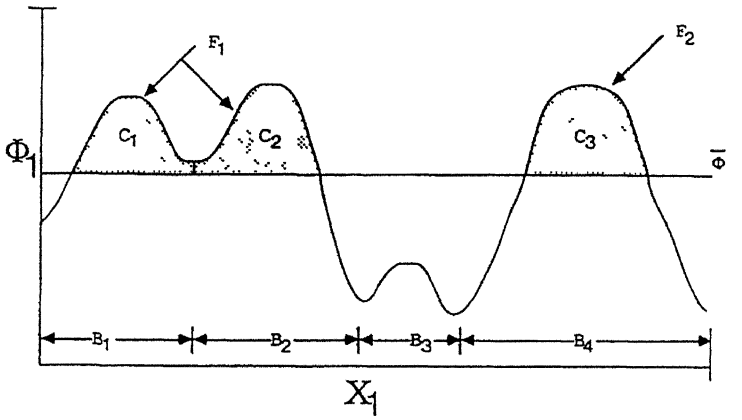


Figure 2 1 A mean slice, its basins, its features and its virtual characters (Taken from Goldberg, 1990)

The processing of the *virtual characters* and *virtual alphabets* can be seen by considering the figure 2 2 Here the variable x_1 is coded on a ternary alphabet and x_2 on a binary alphabet The regions of likely subsequent search are shown as the cross-hatched intersections between virtual characters on different dimensions

Interval Schemata

Eshelman et al in 1993 proposed the concept of *interval schemata* for the analysis of *RCGAs* Since the computers have the limited precision, any real-valued value may be mapped on to an integer So

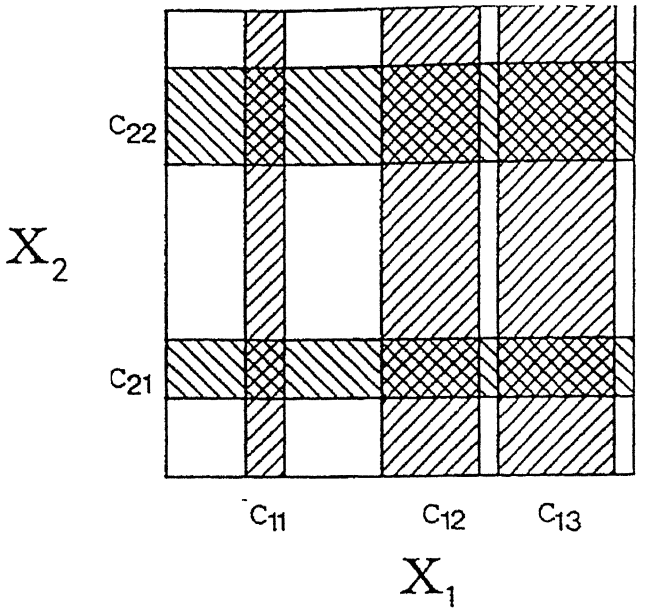


Figure 2 2 The cross-hatched regions are likely to be processed by the algorithms (Taken from Goldberg, 1990)

they suggested the *RCGA* to be called the integer-coded GAs

Analyzing the crossover operators for *RCGAs*, it may be observed that they exploit the intervals determined by the parents rather than the patterns of the symbols they share The *BCGA* were developed for strings of symbols, which is too restrictive for analyzing the *RCGA* Something analogous is needed for *RCGA* that manipulate intervals rather than the bit values From this idea the concept of *interval schemata* arises An *interval schemata*, ι, is defined as

$$\iota = \prod_{i=1}^{n} (\alpha_i, \beta_i)$$

with $\alpha_i, \beta_i \in \{0, \quad 2^{L_i}\}$ and $\alpha_i \leq \beta_i$

The Wright's connected schema and interval schema may be compared by considering a parameter i with range $(0, \quad 2^{L_i})$ For this parameter there are $2^{L_i+1} - 1$ *connected schemata*, however there are $\frac{2^{L_i}(2^{L_i}+1)}{2}$ *interval schemata* which represents a larger quantity Thus *interval schemata* represents connected as well as disconnected intervals

Formae

In (Radcliffe, 1991a) the notion of implicit parallelism is extended (and the associated schema theorem) to general non-string representations through the introduction of arbitrary *equivalence relations* In doing so it provides a framework within which arbitrary genetic operators may be usefully analyzed Schemata are generalized to *formae* which are equivalence classes induced by equivalence relations

over search space The aim is to maximize the predictive power of the schema theorem by allowing the developer of a GA for some particular problem to code knowledge about the search space by specifying families of formae that might reasonably be assumed to group together solutions with related performance and developing operators which manipulate these to good effect Radcliffe presented design principles for building useful equivalence relations The following principle are related to crossover operators

1 **Respect** Crossing two instances of any forma should produce another instance of that forma

2 **Proper Assortment·** Given instances of two compatible formae (e g the ones where there are a chromosome being an instance of both), it should be possible to cross them to produce a child which is an instance of both formae

In (Radcliffe, 1991b) four different types of formae in conjunction with operators for their effective manipulation are discussed The first is related to standard schemata the second to the o-schemata introduced in (Goldberg, 1989a) to manage permutation problems the other two are the *edge formae* defined to deal with chromosomes that represents edges and *locality formae* which relate chromosomes on the basis of their closeness to each other and are applied to problems with continuous parameters

In (Radcliffe, 1991a) the formae to manipulate real coding are discussed again Two properties are considered to be captured by formae namely locality and periodicity The author presents two types of equivalence relations to induce formae describing these properties

Above mentioned are several schemata like concepts developed for real-coded genetic algorithms One common feature of all the concepts discussed above is that all of them represent certain contiguous spaces or intervals in search space These concepts can be used to predict the behaviour of the real-coded genetic algorithms

2 1.2 Crossover Operators

There are a number of the crossover operators available in literature for *RCGA* that have been proposed by many researchers from time to time The crossover operators available for *RCGA* are described in the following subsections, Crossover operators in real-coded Genetic Algorithms *(RCGA)* can be divided in two parts, namely

1 Two Parent Recombination

2 Multi Parent Recombination

2 1 3 Two Parent Recombination

In two parental recombination, two parents participate in crossover and create children Most of crossover operators proposed for *RCGA* are of this type The crossover in multi-variable problems is done using any of the following techniques,

- Line crossover - The children points are generated along the diagonal formed by the parents
- Variable by variable - The crossover is done considering each variable separately

Let us assume $\mathbf{x^1} = (x_1^1, x_2^1, \quad x_n^1)$ and $\mathbf{x^2} = (x_1^2, x_2^2, \quad x_n^2)$ be two parents which have been selected for the crossover from a n-dimensional search space The counter i is running from 1 to n The different crossover operators are described below –

Flat Crossover (Radcliffe, 1991)

This operator produces children points on the diagonal of the hypercube formed by the parents This is a line crossover

An offspring $\mathbf{y} = (y_1, y_2, \quad y_n)$ is generated uniformly from the interval (x_i^1, x_i^2) as

$$y_i = (1 - \alpha)x_i^1 + \alpha x_i^2 \tag{2 1}$$

$\alpha \in (0, 1)$ and remains same for all values of i in one crossover operation This way the children points are being generated along the line formed by joining the two parents, Figure 2 3

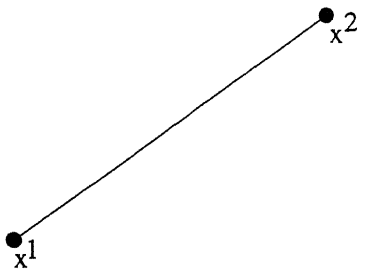


Figure 2 3 The children points are generated along the diagonal formed by parents

This operator produces children points only in the region spanned by the parents

Simple Crossover (Wright, 1991; Michalewicz, 1992)

This crossover operator is same as Single Point Crossover in Binary Coded Genetic Algorithms *(BCGA)* The children created with this operator lie on one of the corners of the hyper-cube formed by the parents excluding the corners occupied by the parents A 2-D case is shown in figure 2 4 Here a cross site

$k \in (1\ 2 \quad n-1)$ is randomly chosen and the right side of two parents are swapped to give two offsprings as

$$\mathbf{y^1} = (x_1^1, x_2^1, \quad x_k^1, x_{k+1}^2 \quad x_n^2) \tag{2 2}$$

$$\mathbf{y^2} = (x_1^2, x_2^2, \quad x_k^2, x_{k+1}^1, \quad x_n^1) \tag{2 3}$$

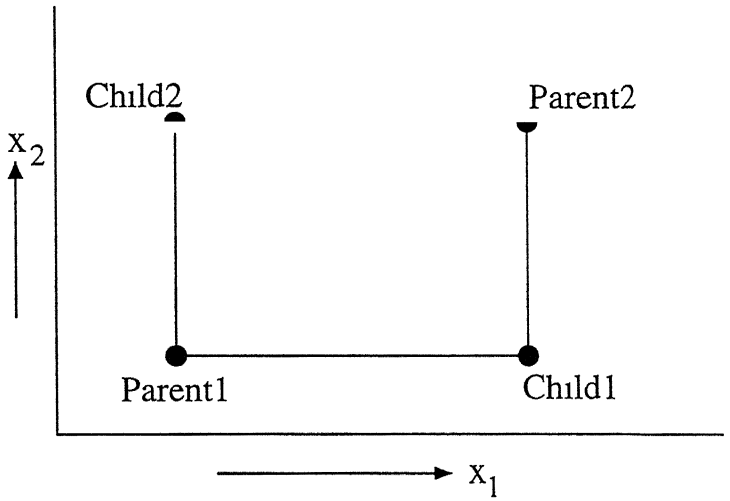


Figure 2 4 Simple Crossover

Arithmetical Crossover (Michalewicz, 1992)

This operator creates children points along the extended diagonal of the n-dimensional hypercube as formed by the parents The children points created are the corner points of the diagonal of the extended hypercube A 2-D case is shown in figure 2 5 If the value of parameter $\lambda > 1\ 0$ the children points will be the corners of the extended hypercube If $\lambda < 1\ 0$ the hypercube will shrink and children points will be corner points of this hypercube

Two offsprings $\mathbf{y^1} = (y_1^1, y_2^2, \quad y_n^1)$ and $\mathbf{y^2} = (y_1^2, y_2^2, \quad y_n^2)$ are generated as

$$y_i^1 = \lambda x_i^1 + (1-\lambda) x_i^2 \tag{2 4}$$

$$y_i^2 = (1-\lambda) x_i^1 + \lambda x_i^2 \tag{2 5}$$

λ is a pre-specified constant This constant λ makes this operator a deterministic operator because for any two given parents we will always get same offsprings since value of λ is fixed

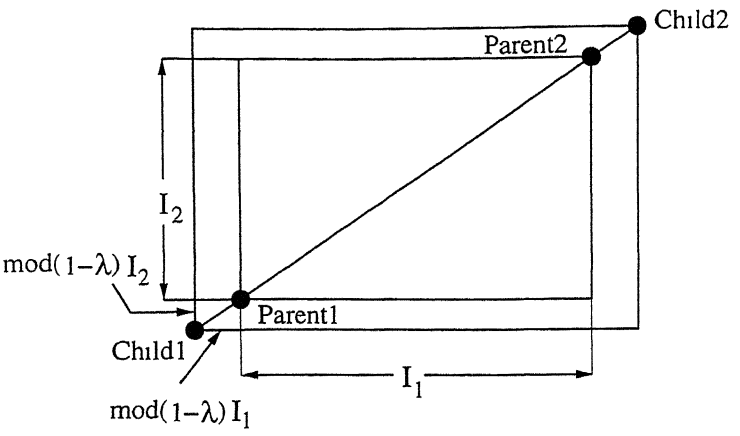


Figure 2 5 Arithmetical Crossover

This operator can be thought of as a discrete case of line BLX-α crossover operator i e in this case we only take the extreme points Depending on the value of λ, the children points can be generated inside the range of parents or outside the parents

BLX-α Crossover (Eshelman et.al., 1993)

This operator creates the offsprings by randomly picking the points from the hypercube obtained after extending the hypercube formed by the parents by a factor of α in all the directions In its present form this operator is applied variable by variable

An offspring $y = (y_1, y_2, \quad y_n)$ is randomly created from the interval defined over $[x_{min} - I\alpha]$ to $[x_{max} - I\alpha]$, where $x_{min} = min(x_i^1, x_i^2)$, $x_{max} = max(x_i^1, x_i^2)$ and I = $x_{max} - x_{min}$ assuming $x_i^1 < x_i^2$ as follows,

$$y_i = (1 - \xi)x_i^1 + \xi x_i^2 \tag{2 6}$$

ξ is uniformly generated random number in the range $(-\alpha, 1 + \alpha)$ This operator picks up points uniformly from a hyper rectangle formed by extending the parents by an amount αI in both sides of a direction A 2-D example is shown in Figure 2 6

Linear Crossover (Wright, 1991)

This operator produces three offsprings $\mathbf{y^1} = (y_1^1, y_2^1, \quad y_n^1)$, $\mathbf{y^2} = (y_1^2, y_2^2, \quad y_n^2)$ and $\mathbf{y^3} = (y_1^3 \; y_2^3, \quad y_n^3)$ from two parents x_i^1 and x_i^2 as

$$y_i^1 = \frac{1}{2}x_i^1 + \frac{1}{2}x_i^2 \tag{2 7}$$

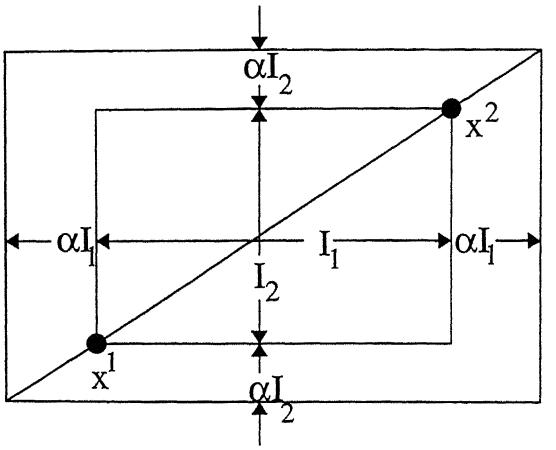


Figure 2 6 BLX-α The children points are randomly uniformly picked from the rectangle formed by the two parents

$$y_i^2 = \frac{3}{2}x_i^1 - \frac{1}{2}x_i^2 \tag{2 8}$$

$$y_i^3 = -\frac{1}{2}x_i^1 + \frac{3}{2}x_i^2 \tag{2 9}$$

The two best offspring replace the parent in the next generation, Figure 2 7

This operator can be considered as a discrete case of *BLX-0 5* as two extreme and one midpoint are being created as the children solutions

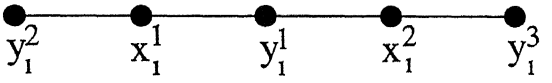


Figure 2 7 Linear Crossover

Discrete Crossover (Muhlenbein et. al., 1993)

This operator is more or less similar to the *simple crossover* i e the offsprings generated by this operator are also the corners of the hypercube formed by all of the points in the parental population The difference between this operator and the *simple crossover* is that here for any dimension the value of the child is randomly selected from the pool of all the values of the parental population of that dimension while in simple crossover two parents are selected and any child is selected from the pool of values of those parents only

An offspring y_i is generated by randomly selecting the component from the set $(x_i^1\ x_i^2)$

Extended Line Crossover (Muhlenbein et al , 1993)

This operator is same as the line version of BLX-0 25

An offspring $\mathbf{y} = (y_1, y_2, \quad y_n)$ is created from two parents as

$$y_i = x_i^1 + \alpha(x_i^2 - x_i^1) \tag{2 10}$$

α is uniformly chosen random number from the range $(-0\ 25, 1\ 25)$ and is same for all $i = 1\ 2 \quad n$

This operator produces offspring on the line formed by joining the parents, Figure 2 8

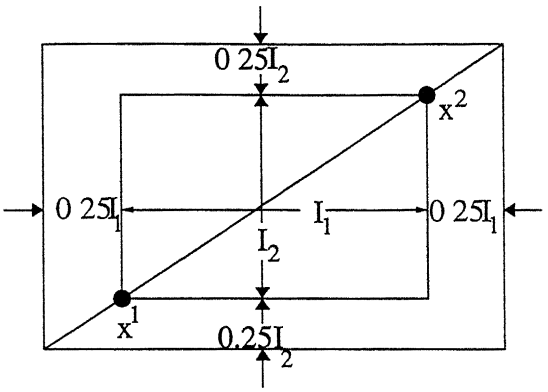


Figure 2 8 Extended Line Crossover Generates points on the extended diagonal formed by parents

Extended Intermediate Crossover (Mühlenbein, 1993)

An offspring $\mathbf{y} = (y_1, y_2, \quad y_n)$ is generated as follows

$$y_i = x_i^1 + \alpha_i(x_i^2 - x_i^2) \tag{2 11}$$

α_i is uniformly generated random number from the range (-0 25, 1 25)

This operator is same as BLX-0 25 The child is randomly picked from this extended hypercube

Wright's Heuristic Crossover(Wright, 1990)

If $\mathbf{x}^1$ is better in fitness than $\mathbf{x}^2$, a single offspring is generated as

$$y_i = \xi(x_i^1 - x_i^2) + x_i^1 \tag{2 12}$$

ξ is uniformly generated number from the range (0,1)

Since this operator produces children points near the parent which is better in fitness so it introduces bias towards the points in the parental population which are better in fitness (Figure 2 9) Thus over generations it tries to shift the population in a certain direction which depends on the fitness landscape

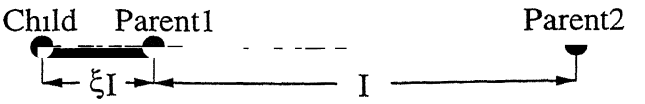


Figure 2 9 Wright's heuristic operator Parent1 is better in fitness



Figure 2 10 Linear BGA crossover The child is randomly picked from the thick portion near first parent

Linear BGA Crossover (Schlierkamp-Voosen, 1994)

This operator is based on Muhlenbein's mutation described in the next section

In this operator an offspring $\mathbf{y} = (y_1, y_2, \quad y_n)$ is created as follows,

$$y_i = x_i^1 \pm rang_i \gamma \wedge \tag{2 13}$$

Here $rang_i = 0\,5(b_i - a_i)$ where b_i and a_i are the upper and lower bounds on the i^{th} variable respectively $\gamma = \sum_{k=0}^{k=15} \alpha_k 2^{-k}$ where $\alpha \in (0, 1)$ is chosen with a probability $\frac{1}{16}$ The -ve sign in the above equation is chosen with a probability 0 9 and $\wedge$ is defined as

$$\wedge = \frac{x_i^2 - x_i^1}{||\mathbf{x^1} - \mathbf{x^2}||}$$

As can be seen, the operator also gives a bias to create the offspring near first parent Figure 2 10

Unfair Average Crossover (Nomura and Miyoshi, 1995)

This operator can be considered as an extension of the single point binary crossover in that here also a cross-site is chose randomly over the length of the individual and the crossover is done differently for the variables falling on left and right side The difference is that here the variables in parent individuals are blended instead of swapped The crossover is done as below.

This operator produces two offsprings $\mathbf{y^1} = (y_1^1, y_2^1, \quad . y_n^1)$ and $\mathbf{y^2} = (y_1^2, y_2^2, \quad y_n^2)$ as follows,

$$y_i^1 = \begin{cases} (1+\alpha)x_i^1 - \alpha x_i^2 & i = 1, 2, \quad j \\ -\alpha x_i^1 + (1+\alpha)x_i^2 & i = j+1, \quad n \end{cases} \tag{2 14}$$

$$y_i^2 = \begin{cases} (1-\alpha)x_i^1 + \alpha x_i^2 & i = 1\,2, \quad j \\ \alpha x_i^1 + (1-\alpha)x_i^2 & i = j{+}1, \quad n \end{cases} \tag{2 15}$$

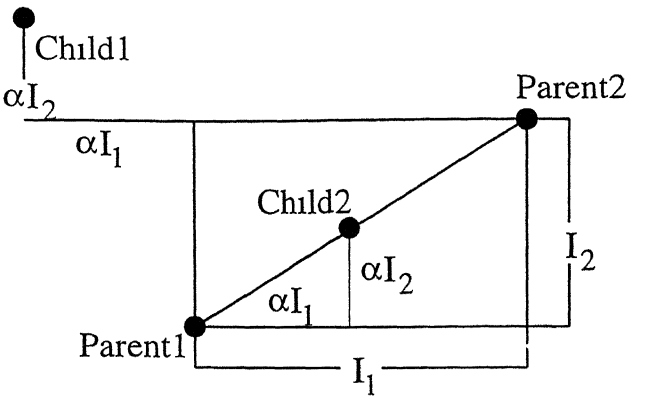


Figure 2 11 Unfair average crossover

Here α is a constant and j is randomly selected from $(0, 1, \quad n)$

The children solutions are positioned according to the value of α selected A 2-dimensional case is shown in figure 2 11 As can be seen, one child is inside the rectangle formed by the parents and other one is outside near first parent If $\alpha < 0\,5$ both of the children will be nearer to the first parent one inside the hyper-cube (in case of multi-dimension problem) and other outside it On the other hand if $\alpha > 0\,5$ both of the children will be away from first parent, one inside the hyper-cube and other outside For $\alpha > 1\,0$, both parents will be outside the hyper-cube

Fuzzy Connective Based Crossover (Herrera et al., 1994)

Here a set of four crossover operators are suggested which generate offsprings in different regions of search space These operators are named F, S, M & L (Figure 2 12) and these are associated with fuzzy connectives T, G, P & $\hat{C}$ respectively These crossover operators generate offspring as

$$F(y_i^1, y_i^2) = a_i + (b_i - a_i)T(s_i^1, s_i^2) \tag{2 16}$$

$$S(y_i^1, y_i^2) = a_i + (b_i - a_i)G(s_i^1, s_i^2) \tag{2 17}$$

$$M(y_i^1, y_i^2) = a_i + (b_i - a_i)P(s_i^1, s_i^2) \tag{2 18}$$

$$L(y_i^1, y_i^2) = a_i + (b_i - a_i)\hat{C}(s_i^1, s_i^2) \tag{2 19}$$

a_i and b_i are lower and upper bounds on the parameter i $s_i^1 = \frac{x_i^1 - a_i}{b_i - a_i}$ and $s_i^2 = \frac{x_i^2 - a_i}{b_i - a_i}$

The fuzzy connective used are as below

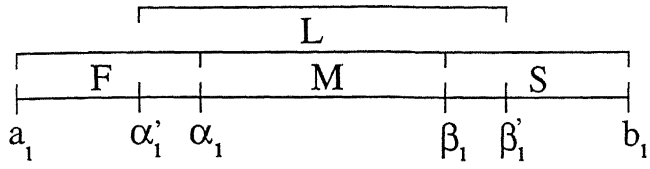


Figure 2 12 Fuzzy Connective Based Crossover Operators

Family	t-norm	t-conorm	Averaging Func. $(0 \leq \lambda \leq 1)$	Gen Comp operator
Logical	$T_1(x,y) = min(x,y)$	$G_1(x,y) = max(x,y)$	$P_1(x,y) = (1-\lambda)x + \lambda y$	$\hat{C}_1 = T_1^{1-\lambda} G_1^{\lambda}$
Hamacher	$T_2(x,y) = \frac{xy}{x+y-xy}$	$G_2 = \frac{x+y-2xy}{1-xy}$	$P_2(x,y) = \frac{1}{\frac{y-y^{\lambda}-xy+x^{\lambda}}{x^{\lambda}}+1}$	$\hat{C}_2 = P_2(T_2\ G_2)$
Algebraic	$T_3(x,y) = xy$	$G_3(x,y) = x+y-xy$	$P_3(x,y) = x^{1-\lambda} y\lambda$	$\hat{C}_3 = P_3(T_3\ G_3)$
Einstein	$T_4(x,y) = \frac{xy}{1+(1-x)(1-y)}$	$G_4(x,y) = \frac{x+y}{1+xy}$	$P_4(x,y) = \frac{2}{1+(\frac{2-x}{x})^{1-\lambda}(\frac{2-y}{y})^{\lambda}}$	$\hat{C}_4 = P_4(T_1\ G_1)$

Table 2 1 Fuzzy Connective Based Crossover Operators

These fuzzy connectives fulfill following properties

$$T_4 \leq T_3 \leq T_2 \leq T_1 \leq P_j (j = 1, \quad 4) \leq G_1 \leq G_2 \leq G_3 \leq G_4$$

As is clear from above table for fuzzy connectives, these crossover operators are *deterministic* because for any given two parents we will always get the same child from a particular operator

Fuzzy Recombination Operator (Voigt et al., 1995)

This recombination operator produces offspring from a triangular probability distribution having a modal values at the parents Let two components of the parents x^1 and x^2 be x_i^1 and x_i^2 respectively then the probability that the offspring has a value y_i is given by a bimodal distribution $p(y_i) \in \{\Phi(x_i^1), \Phi(x_i^2)\}$ with triangular probability distribution $\Phi(r)$ having the modal values x_i^1 and x_i^2 with

$$x_i^1 - d(x_i^2 - x_i^1) \leq r \leq x_i^1 + d(x_i^2 - x_i^1)$$

$$x_i^2 - d(x_i^2 - x_i^1) \leq r \leq x_i^2 + d(x_i^2 - x_i^1)$$

The children points will be calculated as follows,

$$y_i^1 = -\zeta_i d x_i^1 + (1 + \zeta_i d) x_i^2 \tag{2 20}$$

$$y_i^2 = (1 + \zeta_i d) x_i^1 - \zeta_i d x_i^2 \tag{2 21}$$

ζ is a random number following a triangular probability distribution assuming $x_i^1 \leq x_i^2$. Since this operator is generating points using a triangular probability distribution so it has more probability of generating children near parents (Figure 2.13)

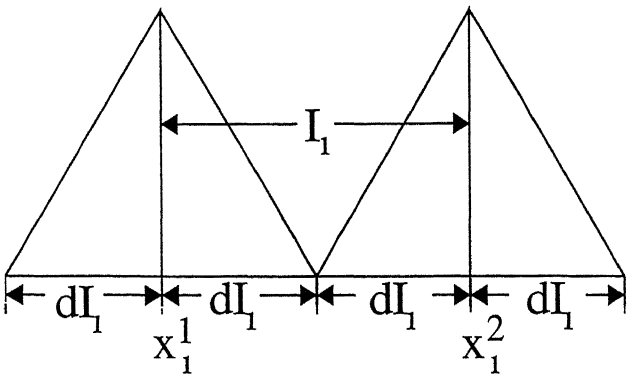


Figure 2.13 Fuzzy Recombination Operator d = 0.5

BLX-α-β (Eshelman, Mathias, Schaffer, 1997)

This crossover operator is again of two types –

1. Box-BLX
2. Directional BLX

1. **Box-BLX** Box-BLX is of two types. These are "Pairwise Recombination" and "Pool-wise Recombination"

 In "Pairwise Recombination Operator" the children points are generated by randomly choosing points from the discretized interval $[x_i^1 - \alpha I_i, x_i^2 + \beta I_i]$ if x^1 is having better fitness than x^2. So if the problem contains n parameters, the offspring will be chosen randomly from the *hyper rectangle* defined by n intervals. Here I_i is $x_i^2 - x_i^1$. A 2-D case is shown in Figure 2.8.

 The "Pool-wise operator" is different from the pairwise operator in that the parents are randomly chosen from the *pool* of values of different components of vector i.e. for i^{th} component two parents will recombine and for j^{th} two different parents may recombine here $(i, j) \in (1\ 2 \quad n)$. Mathematically the recombination can b expressed for both of the above cases as

$$y_i = (1 - \xi)x_i^1 + \xi x_i^2 \tag{2.22}$$

ξ is the uniformly generated random number from the range $(-\alpha, 1+\beta)$ α and β are user defined constants In case of "Pairwise Recombination" $\forall i = 1, 2, \ldots n$ two parents recombine In case of Pool-wise Recombination" two different parents are chosen for every $i = 1, 2 \ldots n$

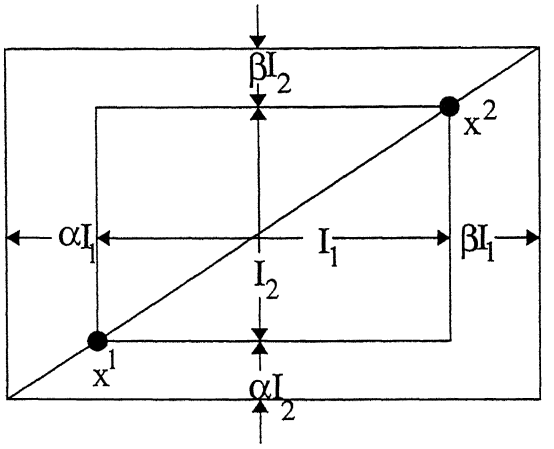


Figure 2 14 Box BLX-α-β

2 **Directional BLX** In this case the *hyper rectangle* is trimmed by a factor of γI_i in each direction and the offspring is randomly chosen from this trimmed *hyper rectangle* (Figure 2 9)

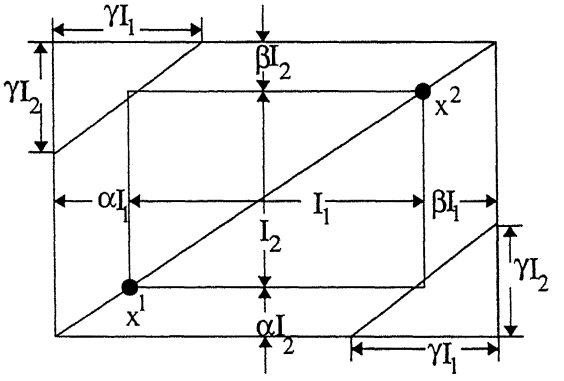


Figure 2 15 Directional BLX-α-β, The children are uniformly randomly selected from the trimmed rectangle

Simulated Binary Crossover (Deb et.al , 1995)

This operator simulates the working of *binary single point crossover* It works without the coding of the problem parameters i e on real numbers To achieve that the *search power* of the crossover operator is defined in terms of the probability distribution of any child string to be created from any two given parent strings The SBX has the search power similar to that of the single point crossover Here

a parameter called "spread factor', denoted by β is defined Based on this parameter the crossover operator can be of three types Operators having $\beta < 1$ are called *contracting* operators and those having $\beta > 1$ are called *expanding* operators The operators having $\beta = 1$ are termed as *stationary* operators The random variate β is generated using the polynomial distribution (Figure 2 10) as below

$$p(\beta) = 0\ 5(\eta + 1)\beta^{\eta}, \text{for } \beta \leq 1 \tag{2 23}$$

$$p(\beta) = 0\ 5(\eta + 1)\frac{1}{\beta^{\eta+2}}, \text{for } \beta > 1 \tag{2 24}$$

The value of "spread factor is calculated by equating the area given by a randomly generated random number between 0 and 1 to above equations Here η is the distribution index of the probability distribution After calculating β from above equations the children points are generated as

$$y_i^1 = 0\ 5\left[(1 + \beta_i)x_i^1 + (1 - \beta_i)x_i^2\right] \tag{2 25}$$

$$y_i^2 = 0\ 5\left[(1 - \beta_i)x_i^1 + (1 + \beta_i)x_i^2\right] \tag{2 26}$$

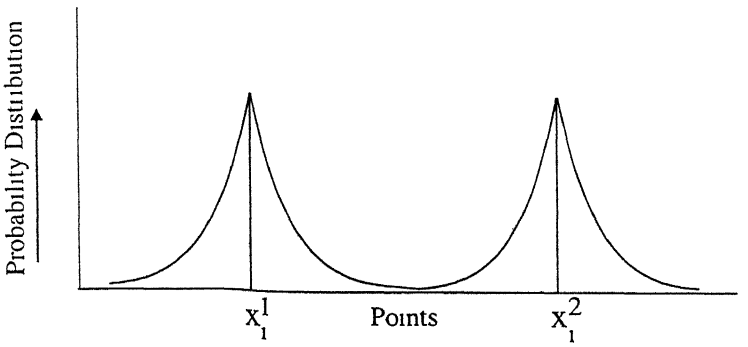


Figure 2 16 SBX Crossover Operator Probability Distribution for the generation of Children

Here β_i is defined as $\beta_i = |\frac{y_i^2 - y_i^1}{x_i^2 - x_i^1}|$ The spread factor is randomly generated from a polynomial distribution defined above for each variable in case of "variable by variable" crossover

The β is generated by transformation of above polynomial distribution using uniform variate as follows

$$\beta(u) = \begin{cases} (2u)^{\frac{1}{\eta+1}}, & \text{if } u(0,1) \leq 0\ 5 \\ [2(1-u)]^{-\frac{1}{\eta+1}}, & \text{if } u(0,1) > 0\ 5 \end{cases} \tag{2 27}$$

u is a uniformly generated random number between 0 and 1

In case of multi-variable problems, the crossover is done "variable by variable" i e for a variable the crossover will take place with a probability 0 5 If the crossover is to be done the value of β is calculated using above equations

The characteristic of the SBX operator is that it generated offsprings depending on the position of the parents If the parents are far, the children solutions are created far away and if the parents are close the children are generated close Also a small value of the distribution index creates points far away while a larger value gives closer solutions

Simulated binary crossover for problems having no bounds on the variables has been described above If the problem parameters have bounds, SBX can be used after some modification In this case the area of the probability distribution which lies outside the bounds is added to that is enclosed by the bounds Mathematically it is done as follows,

$$\beta_q = \begin{cases} (u\alpha)^{\frac{1}{\eta+1}}, & \text{if } u \leq \frac{1}{\alpha} \\ \left(\frac{1}{2-u\alpha}\right)^{\frac{1}{\eta+1}}, & \text{otherwise} \end{cases} \tag{2 28}$$

where $\alpha = 2 - \beta^{-(\eta+1)}$ and β is calculated as

$$\beta = 1 + \frac{2}{x_2 - x_1} min\left[(x_1 - x_l), (x_u - x_2)\right]$$

Here, x_l and x_u are the lower and upper bounds on the parameter x

2.1.4 Multi Parent Recombination

In Multi Parent Recombination more than two parents take part in recombination Let M be the number of parents taking part in recombination Let $\mathbf{x^k} = (x_1^k, x_2^k, \quad x_n^k)$ represents the parent vector where k is counter running from 1 to M Let i be a counter running from 1 to n n is the number of variables The different multi parent recombination operators are presented below –

Unimodal Normal Distribution Crossover (Ono and Kobayashi, 1997)

This operator uses three parents to create two offsprings Let $\mathbf{x^1}, \mathbf{x^2}$ and $\mathbf{x^3}$ be the parents & $\mathbf{y^1}$ and $\mathbf{y^2}$ be the offsprings Here $\mathbf{x^k}$ and $\mathbf{y^k}$ represents n-dimensional vectors

The offsprings are generated by using the *normal distribution* which is defined by three parents Two different *normal distribution* are used, one is along the line connecting first two parents i e $\mathbf{x^1}$ and $\mathbf{x^2}$ whose *standard deviation* is proportional to the distance between the two parents The other one is along the line connecting third parent and the line defined by first two parents and the *standard deviation* of this distribution is proportional to the *perpendicular distance* of third parent from the line defined by first two parents (Figure 2 11)

Mathematically the procedure is described below –

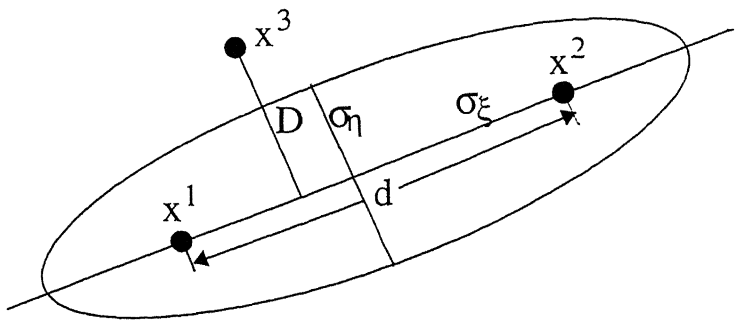


Figure 2 17 UNDX Operator

$$\mathbf{y^1} = \mathbf{x_p} + \xi \mathbf{d} + \sum_{j=1}^{n-1} \eta_j D \mathbf{e_k} \tag{2 29}$$

$$\mathbf{y^2} = \mathbf{x_p} - \xi \mathbf{d} - \sum_{j=1}^{n-1} \eta_j D \mathbf{e_k} \tag{2 30}$$

where, $\mathbf{x_p}$ is the mean vector of first two parents i e $0\,5(\mathbf{x^1} + \mathbf{x^2})$, ξ is the randomly generated number following *normal distribution* $N(0, \sigma_\xi^2)$, η_j are $(n-1)$ randomly generated numbers following *normal distribution* $N(0, \sigma_\eta^2)$, $\mathbf{d}$ is the difference vector of $\mathbf{x^1}$ and $\mathbf{x^2}$ i e $(\mathbf{x^2} - \mathbf{x^1})$ and e_k are $(n-1)$ orthonormal bases that span subspace perpendicular to formed by first two parents and follow $\mathbf{e_j} \perp \mathbf{e_l}$ where $(j, l) \in (1, 2, \quad n-1)$ and $j \neq l$

This operator generates offspring which are symmetrically placed with respect to the mean of first two parents

Multi Parental Unimodal Normal Distribution Crossover (Kita, Ono and Kobayashi, 1999)

This crossover operator is an extension of *three parental UNDX* Here any number of parents between 2 and $n+1$ can participate in the crossover

Here m parents take part in crossover and two children are generated The children points are generated as below,

1 Choose $m+2$ parents randomly from the population Let these be $\mathbf{x^1}, \mathbf{x^2} \quad \mathbf{x^{m+2}}$

2 Calculate the centre of mass of first $m+1$ parents as

$$\mathbf{x_p} = \frac{1}{m+1} \sum_{i=1}^{i=m+1} \mathbf{x^i}$$

Calculate the difference vector of $\mathbf{x^i}$ and $\mathbf{x_p}$ as,

$$\mathbf{d^i} = \mathbf{x^i} - \mathbf{x_p}$$

3 Let D be the length of component of $\mathbf{d^{m+2}} = \mathbf{x^{m+2}} - \mathbf{x_p}$ orthogonal to $\mathbf{d^1}, \mathbf{d^2}$ $, \mathbf{d^{m+1}}$

4 Let $\mathbf{e^1}, \mathbf{e^2}$ $, \mathbf{e^{n-m}}$ be the orthonormal bases of the subspace orthogonal to the subspace spanned by $\mathbf{d^1}$ $\mathbf{d^2}$ $, \mathbf{d^{m+1}}$

5 Generate the children points as follows,

$$\mathbf{y^1} = \mathbf{x_p} + \sum_{k=1}^{m} \xi_k \mathbf{d}^k + \sum_{k=1}^{n-m} \eta_k D \mathbf{e}^k \tag{2 31}$$

$$\mathbf{y^2} = \mathbf{x_p} - \sum_{k=1}^{m} \xi_k \mathbf{d}^k - \sum_{k=1}^{n-m} \eta_k D \mathbf{e}^k \tag{2 32}$$

Here ξ_k and η_k are the randomly generated number following the normal distributions $N(0\ \sigma_{1l}^2)$ and $N(0\ \sigma_\eta^2)$ respectively

Simplex Crossover (Tsutsui et.al 1999)

The simplex crossover operator (SPX) creates a single offspring from $n+1$ parents Here n is the number of variables In the crossover operation these $n+1$ parents form a simplex This simplex is extended by a factor of ϵ with respect to the centre of mass of the parents The child is created by randomly and uniformly picking the point from this extended simplex The procedure of creating the child is described below,

1 Choose $n+1$ parents from the parental population randomly Let the parents be $\mathbf{x^0}, \mathbf{x^2}$, $, \mathbf{x^n}$

2 Calculate the centre of mass of the parents chosen above as,

$$\mathbf{x_p} = \frac{1}{n+1} \sum_{j=0}^{n} \mathbf{x^j}$$

3 Generate r_k, which is random number given by,

$$\mathbf{r_k} = (u(0,1))^{\frac{1}{k+1}} \text{ for } k = 0, \quad , n-1$$

Here $u(0\ 1)$ is uniformly generated random number between 0 and 1

4 Calculate $\mathbf{X_k}$ and $\mathbf{C_k}$ as below,

$$\mathbf{X_k} = \mathbf{x_p} + \epsilon \left(\mathbf{x^k} - \mathbf{x_p} \right)$$

$$\mathbf{C_k} = \begin{cases} 0 & \text{for } k = 0 \\ \mathbf{r_{k-1}} (\mathbf{X_{k-1}} - \mathbf{X_k} + \mathbf{C_{k-1}}) & \text{for } k = 1, \quad , n \end{cases}$$

where ϵ is the expansion rate

5 An offspring is given by,

$$C = \mathbf{X_n} + \mathbf{C_n} \tag{2 33}$$

The working of SPX operator is shown in figure 2 18 for a 2-dimensional case

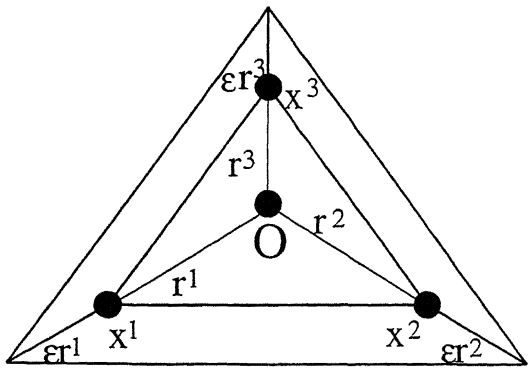


Figure 2 18 The SPX Operator The children points are chosen from within the extended simplex

2 1 5 Mutation Operators

Although we have not analyzed the mutation operators here, we describe a few commonly used in the following

Let $\mathbf{x} = (x_1, x_2, \quad x_i, \quad x_n)$ be an individual where n is number of parameters x_i represents the i^{th} variable to be mutated and x_i' be the mutated gene The different mutation operators are presented below.

Random Mutation (Michalewicz, 1992)

Here r_i' is a uniformly generated random number from the range $[a_i, b_i]$ where a_i and b_i are the lower and upper bounds on the i^{th} variable

Non-uniform Mutation (Michalewicz, 1992)

Let g be the number of generation at which the mutation is to be carried out and g_{max} be the maximum number of generations, the mutated variable is given by,

$$x_i' = \begin{cases} x_i + \Delta(g, b_i - x_i) & \text{if } \tau = 0 \\ x_i - \Delta(g, x_i - a_i) & \text{if } \tau = 1 \end{cases} \tag{2 34}$$

Here τ is a random number which may have a value of zero or one and the operator Δ is defined as

$$\Delta(g, y) = y\left(1 - r^{(1-\frac{g}{g_{max}})^b}\right)$$

Here r is uniformly generated number from the range (0,1) and b is a parameter chosen by the user that determines the dependency on the number of iterations This function gives a value in the range $[0, y]$ such that the probability of returning a number close to zero increases as the algorithm advances The size of gene generation interval should become smaller with the number of generations

This property causes this operator to make a uniform search in the initial space when g is small and very locally at a later stage

Real Number Creep (Davis, 1991)

In this operator the chromosome is made to slide into a value which increases or decreases it by a small random quantity This is advantageous in case the individual is near some maxima or minima The maximum slide allowed is determined by a parameter set by user Different instances of this operator have been proposed, such as Guaranteed Big Creep and Guaranteed Little Creep (Davis, 1989) and small creep and large creep The difference between these operators lies in the value of the maximum slide allowed

Muhlenbein's Mutation (Mulenbein et. al., 1993)

The mutated gene is given as

$$x_i' = x_i \pm rang_i \gamma \tag{2.35}$$

where $rang_i$ defines the mutation range and it is normally set to $0.1(b_i - a_i)$ The + or - sign is chosen with a probability 0.5 and γ is defined as

$$\gamma = \sum_{k=0}^{k=15} \alpha_k 2^{-k}$$

Here $\alpha_i \in (0, 1)$ is randomly generated with probability of $\frac{1}{16}$

Discrete Modal Mutation (Voigt et.al., 1994)

This is a generalization of the above operator It differs only in the value of γ which is given by following equation

$$\gamma = \sum_{k=0}^{\pi} \alpha_k B_m^k$$

with $\pi = \frac{\log(rang_{min})}{\log(B_m)}$ $B_m > 1$ is a parameter called the base of the mutation and $rang_{min}$ is the lower limit of the relative mutation range

Continuous Modal Mutation (Voigt et.al., 1994)

This operator is also same as Muhlenbein's operator with the difference in the γ, given as

$$\gamma = \sum_{k=0}^{\pi} \alpha_k \phi\left(B_m^k\right)$$

$\phi(z_k)$ is a triangular distribution with $\frac{B_m^k - B_m^{k-1}}{2} \leq z_k \leq \frac{B_m^{k+1} - B_m^k}{2}$

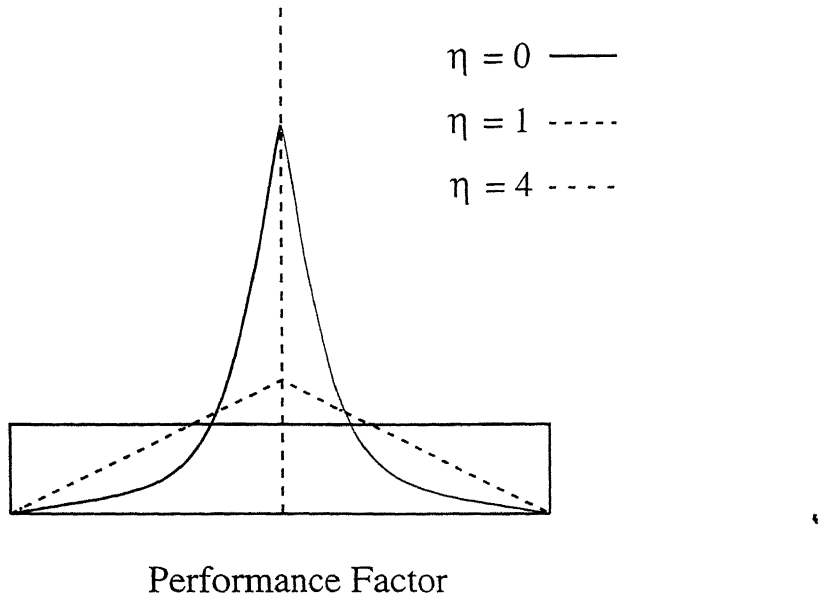


Figure 2 19 Probability distribution for creating a mutated value for continuous variables

Deb's Mutation

In this mutation operator the current value of the variable is changed to a neighboring value using a polynomial distribution having its mean at current value (Fig 2 13) and its variance as a function of the distribution index η To perform the mutation a perturbance factor δ is defined as

$$\delta = \frac{c - p}{\Delta_{max}}$$

here Δ_{max} is a fixed quantity, representing the maximum permissible perturbance in the parent value p and c is the mutated value Similar to the *simulated binary crossover operator* proposed by Deb the mutated value is calculated with a probability distribution that depends on the perturbance factor δ and is given as

$$p(\delta) = 0\ 5(\eta + 1)(1 - |\delta|)^{\eta}$$

The above probability distribution is valid in the range $\delta \in (-1\ 1)$

To create a mutated value, a random number u is generated in the range (0 1) Thereafter the following equation can be used to calculate the perturbance factor $\bar{\delta}$ corresponding to u using the above distribution,

$$\bar{\delta} = \begin{cases} (2u)^{\frac{1}{\eta+1}} - 1 & \text{if } u \leq 0\ 5 \\ 1 - [2(1-u)]^{\frac{1}{\eta+1}}, & \text{otherwise} \end{cases} \qquad (2\ 36)$$

hereafter, the mutated value is calculated as follows,

$$c = p + \overline{\delta}\Delta_{max}$$

2 Variants of Real-coded Genetic Algorithm

In this section some *RCGAs* are described which are some modification of the original real-coded genetic algorithm The different algorithms are as follows

2 2 1 Genetic Adaptive Search (GeneAS), (Deb et al , 1996)

The binary coded genetic algorithm (*BCGA*) discretizes the search space according to string length chosen to represent the problem vector i e accuracy required On the other hand the real-coded genetic algorithm (*RCGA*) work with the representation of problem vector using real numbers The *BCGA* thus discretizes the search space even if the search space is continuous The *RCGAs* in solving problems having discrete search space (in the case of zero-one or discrete variables), assume the search space to be continuous and then introduce artificial constraints to favor permissible discrete values

The GeneAS combines both of these algorithms Here a mixed coding scheme is used to code all zero-one and discrete variables in binary strings and the continuous variables are coded directly The different components of GeneAS are stated below,

Coding

The problem parameters in the parameter vector are coded in either binary strings or directly depending on the type of the variable A zero-one variable is coded in a one bit ($0 or 1$) sub-string A discrete variable is coded in either binary string or directly represented by real number depending on whether the total number of permissible choices for the variable is 2^k or not A continuous variable is coded directly A typical individual with this coding scheme would be,

(1) 15 23 457 (1011)

The first variable is zero one type, second and third one are continuous and the last one is a discrete variable

Selection

The selection operator works same as in the simple genetic algorithms i e the individual having more fitness is selected to be placed in the mating generation Since this operator only depends on the fitness of the solutions, a standard proportionate, ranking or tournament selection can be used

Crossover

Here the crossover is being done variable by variable and a variable is crossed with a probability 0.5. If the variable to be crossed is coded in binary string, a binary crossover operator is applied. On the other hand, if the variable is coded directly a real coded crossover operator as simulated binary crossover (SBX) is used.

Mutation

After the crossover operator is used with a small probability. If the variable selected for the mutation is a binary string, a random bit is changed from 1 to 0 or vice-versa. For the continuous variable the current value of the variable is changed to a neighboring value using a polynomial distribution having its mean at the current value and its variance as a function of the distribution index n as stated above. If the chosen variable is discrete it is changed to a neighboring permissible value with a discrete probability distribution.

2.2.2 BUGS (Iba et al., 1993)

Another algorithm that implements the real coding and uses the concept of the *RCGA* is BUGS proposed by Iba. The main features in which this algorithm differs from the conventional *RCGA* are as follows.

1. The first main difference is in the contents of the individuals. In a conventional *RCGA* an individual is a collections of the values of problem parameters. In this algorithm the population consists of structures called *bugs* which are characterized as.

$$bugs_i(g) = \begin{cases} Position & \mathbf{x}^i(g) = (x_1^i(g) \quad , x_n^i(g)) \\ Direction & \mathbf{dx}^i(g) = (dx_1^i(g), \quad , dx_n^i(g)) \\ Energy & \mathbf{e}^i(g) \end{cases} \tag{2.37}$$

 Here i is a counter running from 1 to population size, n is the number of parameter and the energy of the i^{th} bugs is defined as

$$\mathbf{e}_i = \sum_{k=0}^{\tau_u} f\left(x_i\left(g-k\right)\right) \tag{2.38}$$

 The above equation gives the cumulative sum of the fitness of the positions in the $bugs_i$ over the previous τ_u time steps generations, where τ_u is an input parameter. The position at generation g are updated as

$$\mathbf{x}^i(g+1) = \mathbf{x}^i(g) + \mathbf{dx}^i(g)$$

2 In a conventional *RCGA*, the fitness of an individual is taken to be the objective function value at that particular point, here the fitness of a *bugs* is defined as the energy of the *bugs* i e a *bugs* having more energy is considered to be more fit

3 Another difference is in the application of the crossover and mutation operators The crossover and mutation operators in this algorithm act on the direction component of a *bugs* instead of the position as is done in a *RCGA*

The steps in the algorithm are described below

1 Generate an initial population at random

$$Pop(0) = (bugs_1(0), \quad , bugs_\mu(0))$$

Input τ_u and set $g = 1$ Here μ is the population size

2 For $i = 1, 2 \quad \mu$, do

$$\mathbf{x}^{\mathbf{i}}(g+1) = \mathbf{x}^{\mathbf{i}}(g) + \mathbf{dx}^{\mathbf{i}}(g)$$

3 For $i = 1\ 2, \quad mu$, do

$$\mathbf{e}^{\mathbf{i}}(g) = \mathbf{e}^{\mathbf{i}}(g-1) + \mathbf{x}^{\mathbf{i}}(g)$$

4 If g is a multiple of τ_u, then

- Set $m = 1$
- Select two parents $bugs_i(g)$ and $bugs_i(g)$ using a probability distribution over the energies of all the *bugs* in $Pop(g)$ so that bugs with higher energy are selected more frequently
- With probability p_c apply the crossover operator to the $\mathbf{dx}^{\mathbf{i}}(g)$ and $\mathbf{dx}^{\mathbf{j}}(g)$ forming two offsprings $bugs_m(g+1)$ and $bugs_{m+1}(g)$ in $Pop(g+1)$
- Apply the mutation operator to $\mathbf{dx}^{\mathbf{i}}(g)$ and $\mathbf{dx}^{\mathbf{j}}(g)$ with probability p_m
- set $n = n + 2$
- if $n < N$, goto step 2

5 If g is not a multiple of τ_u then

- $Pop(g+1) = Pop(g)$

- $g = g + 1$
- goto step 2

6 For $i = 1, 2, \ldots, \mu$, do $\mathbf{e}^1(g) = 0$

- $g = g + 1$
- goto step 2

2.3 Evolution Strategies

The evolution strategies were first introduced by Schwefel and Rechemberg. The first *ES* was a two membered algorithm which worked by creating one n-dimensional vector of object variables from its parent by applying mutation with identical standard deviations to each object variable. This simple *ES* is termed as (1+1)-ES. The first multi-membered *ES* or $(\mu+1)$-ES was designed by Rechemberg. In this algorithm μ parents recombine to give one offspring which after mutation replaces the worst parent if the offspring is better. This introduced the concept of population in evolution strategies. Later on more efficient *ESs* were introduced by Schwefel. These include $(\mu + \lambda)$-ES and (μ, λ)-ES. In both of these μ parents recombine to give λ offsprings. In the former algorithm the after mutation next generation is formed by choosing μ best individuals from the union of parents and offsprings while in the later after mutation μ best offsprings form the next generation (i.e. $\lambda > \mu$ is necessary).

There are two major differences between real-coded genetic algorithms and evolution strategies as follows:

1. In *RCGA* the crossover operator is the main search operator while in *ES* the mutation operator is the main search operator and crossover works as a secondary operator.
2. In *RCGA* the object vector consists only of the problem parameter while in a *ES* the object variable includes problem parameters as well as standard deviations also termed as *mutation strength* and rotation angles.

The main features of the *ES* are described below,

2.3.1 Representation

In *ES* the object variable component $\vec{a}$ consists of the problem parameter vector $\vec{x}$ and strategy parameters standard deviation vector, $\vec{\sigma}$ and rotation angle vector $\vec{\alpha}$ i.e. $\vec{a} = (\vec{x}, \vec{\sigma}, \vec{\alpha})$.

Along with the object variable vector $\vec{x}$, each individual has $n_\sigma \in (1, \ldots, n)$ different standard deviations $\sigma_i (i = 1, \ldots, n_\sigma - 1)$ for the normal distributions used and $n_\alpha \in (0, \ldots, \frac{(2n-n_\sigma)(n_\sigma-1)}{2})$ rotation angles

$\alpha_{ij} \in [-\pi, \pi]$ Here n is number of problem parameters and $j \in (i+1, \quad n)$ So in case of $n_\sigma = n$ there will be $\frac{n(n-1)}{2}$ rotation angles giving total strategy parameters to be $\frac{n(n+1)}{2}$ In case $1 < n_\sigma < n$ the standard deviations $\sigma_1, \quad \sigma_{n_\sigma - 1}$ are coupled with object variables $x_1, \quad x_{n_\sigma - 1}$ and σ_{n_σ} is used for the remaining variables

The number of rotation angles depends on the n_σ, the number of standard deviations and n but it can also be explicitly set to zero indicating that this strategy parameter part of the individuals is not used The *ES* without this strategy parameter is called the *uncorrelated ES* while one involving this parameter is known to be *correlated ES*

2 3 2 Mutation

As already pointed out that mutation is the main search operator in the working of an evolution strategy The mutation is applied to an individual $\vec{a}$ as follows,

$$\sigma_i' = \sigma_i \; exp\left(\tau' \; N(0,1) + \tau \; N_i(0,1)\right) \tag{2 39}$$

$$\alpha_j' = \alpha_j + \beta \; N_j(0,1) \tag{2 40}$$

$$\vec{x}' = \vec{x} + \vec{N}(0, C(\vec{\sigma}', \vec{\alpha}')) \tag{2 41}$$

The above equations hold $(\forall i \in \{1, \quad n\}, \; \forall j \in \{1, \quad n\,(n-1)/2\})$ $N(0,1)$ is the realization of a normally distributed one-dimensional random number having mean zero and standard deviation one $N_i(0\;1)$ indicates that the random number is sampled anew for each value of i Similarly $N_j(0\;1)$ is sampled anew for each possible value of j σ_i', α_j' and $\vec{x}'$ denote the values of mutated standard deviation mutated rotation angles and mutated value of problem parameter vector respectively

Multiplication by such a matrix performs a coordinate transformation with respect to axes number i and j and angle α_{ij} Since n_α rotations are needed to represent all correlations, the complete relation between $\vec{\sigma}_c$ and $\vec{\sigma}_u$ results as

$$\vec{\sigma}_c = \left(\prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \mathbf{R}(\alpha_{ij})\right) \vec{\sigma}_u \tag{2 42}$$

The factors τ, τ' and β are other parameters of which the values are suggested as

$$\tau \propto \left(\sqrt{2\sqrt{n}}\right)^{-1}$$

$$\tau' \propto \left(\sqrt{2n}\right)^{-1}$$

$$\beta = 0\ 0873$$

Usually, the values of proportionality constants for the τ and τ' are taken as one and the value of β is suggested to be 5°

2 3 3 Recombination

The recombination in evolutionary strategies can be done using any of the following operators which are in use These are,

- Discrete Recombination
- Intermediate Recombination
- Generalized Intermediate Recombination

Each of the above operators can be of *sexual* or *panmictic form* In the *sexual* form the recombination operators act on two individuals randomly chosen from the population In the *panmictic* form one parent is kept fixed while for each parameter of the parameter vector the other parent is chosen randomly from the population

The different operators used are used below,

Discrete Recombination

In discrete recombination, for each component of the vectors it is decided at random from which of the both parents (in sexual form) the component is copied to the offspring individual The panmictic variant chooses for each component a new parent individual from the whole population

Intermediate Recombination

Intermediate recombination indicates that the offspring's components are obtained by calculating the arithmetic mean of the corresponding components of both parents For the panmictic version one parent is kept fixed while for each component the other parent is chosen at random from the population

Generalized Intermediate Recombination

In this operator the recombination is done using arbitrary weight factors from the range [0 1] besides the value 0 5 as used in intermediate recombination

Mathematically all these operators can be stated as follows,

Let x_1 and x_2 be two parents chosen for recombination and x' be the offspring vector then $\forall i = 1 \quad n$

$$x'_i = \begin{cases} x_{1,i} \text{ or } x_{2,i} & \text{discrete} \\ x_{1,i} \text{ or } x_{2,i_i} & \text{panmictic discrete} \\ x_{1,i} + \frac{(x_{2,i} - x_{1,i})}{2} & \text{intermediate} \\ x_{1,i} + \frac{x_{2,i_i} - x_{1,i}}{2} & \text{panmictic intermediate} \\ x_{1,i} + \chi(x_{2,i} - x_{1,i}) & \text{generalized intermediate} \\ x_{1,i} + \chi(x_{2,i_i} - x_{1,i}) & \text{panmictic generalized intermediate} \end{cases}$$

Here i_{2,i_i} indicates that the second parent is to be sampled anew for each value of the i

2 3 4 Selection

The selection in the evolution strategies is totally deterministic The evolution strategies are represented by notations like $(\mu+\lambda)$-ES, (μ, λ)-ES The former selects best μ individuals from the union of the parents and children The later shows that best μ offsprings will be selected from the λ offsprings

Combining the different steps as discussed above the algorithm for a $(\mu + \lambda)$-ES and a $(\mu \ \lambda)$-ES can be written as

1 Set generation number counter g = 0

2 Randomly initialize the initial population, $P(0) = \{\vec{a}_1(0), \quad , \vec{a}_\mu(0)\}$ Here $\vec{a} = (\vec{x}, \vec{\sigma} \ \vec{\alpha})$

3 evaluate the fitness of the individuals fitness = $\phi(\vec{a})$

4 recombine, mutate and evaluate the fitness of the individuals

5 select the μ individuals according to strategy used

6 if max($\phi(\vec{a})$)-min($\phi(\vec{a})$) $< \epsilon$ terminate
else set g=g+1 and goto step 4

Here μ denotes the population size

2.3.5 Invariance Property of Evolution Strategies

Recently concept of *invariance* has been proposed by Nikolaus Hansen (2000) to evaluate the performance of the evolution strategies The *invariance* concept tries to explore the self adaptive power of the *ES* by finding a certain *ES* performs similarly in transformed problem space or by achieving invariance against certain transformation of the search space Starting from the best strategy parameter setting, with adaptation tuned on the invariance ensures a predictable behaviour of the strategy In the following the *invariance* concept is described and the invariance properties of certain algorithm are discussed

Invariance properties of a search algorithm with respect to changes of the objective function are extremely attractive They raise the probability to get the similar or even identical results on similar not yet evaluated objective functions and increase the predictability of the strategy behaviour Mainly the invariance against the following changes are desirable,

1 **Translation** On $f \quad x \mapsto q(x-a)$ invariance against $a \in R$ This presents the invariance against the linear transformation of any dimension by a constant value i e if the objective function is linearly transformed the algorithm should perform similarly as on the untransformed function

2 **Rotation** On $f \quad x \mapsto q(Ux)$ invariance against the orthogonal matrix U This means the invariance against the chosen (orthonormal) coordinate system and includes the invariance against the rotation and reflection of the search space

Here q is chosen arbitrarily The *invariance* can be lost if more complex operators are introduced in the simple *ES*

If a certain *ES* is able to preserve its performance, it is considered to be good algorithm Hansen analyzed the $(1+\lambda)$-ES against the linear and rotational transformations He used the *sphere* model and its linearly transformed problems and the problems obtained using a orthonormal matrix to test against the rotation He found the $(1+\lambda)$ was not able to perform in the same manner as in the untransformed case i e this ES was not able to retain these invariances and also the performance was too much dependent on the initial guess so a good guess initial guess is also important

The concept of *invariance* can be extended to other *ESs* as well as to other algorithms

2.4 Differential Evolution

Differential Evolution was proposed by Price and Storn (1997) In *DE* also there are selection crossover and mutation but these are different from that in the genetic algorithm or evolution strategy The working of *DE* and its different components are described in the following,

The parallel version of *DE* holds two arrays each of which hold a population of size μ, n-dimensional real valued vector One array named as primary array holds the current vector population while the secondary array contains the vectors which are selected for the next generation

In each generation μ competition are held to determine the next generation The fitness here is termed as cost which is nothing but the objective function value at that particular point or individual In all the *DE* depends on the three factors namely, the population size,μ, the weight applied to the random differential,F, and the constant that mediates the crossover operation, CR The steps in the differential evolution are described below,

2 4 1 Initialization

Each parameter in every primary array vector is initialized with a uniformly distributed random value between the bounds on the variables To determine and preserve the cost i e fitness of the resulting initial population, each primary array vector is evaluated and the results are stored in the array $cost[]$

2.4 2 Mutation

The mutation in the differential evolution is carried out by perturbing a randomly selected vector from the population by adding the difference between two other randomly selected vectors (the difference has been termed as the "vector differential") Therefore if the vector X_c has been selected for the mutation two other vectors X_a and X_b are selected from the population and the vector differential is given by $X_a - X_b$ If the mutated vector is represented by X'_c, the mutation is given as

$$X'_c = X_c + F\ (X_a - X_b) \tag{2 43}$$

Here F is the user specified weight applied to the random differential The value of F lies in the range (0 1 2)

2 4 3 Recombination

Recombination in DE is done using the non-uniform crossover Each primary vector (called the target vector) is to undergo the crossover with another randomly chosen vector Let X_i be the chosen target vector and X'_i be the another parent which has been chosen randomly from rest of the population for mating with the target vector The randomly chosen vector is recombine the target vector after the randomly chosen vector has been mutated Which parameter of the target vector or of randomly chosen vector will form the offspring vector is determined as follows,

Starting from a randomly selected parameter out of the n parameters for every parameter a uniformly generated random number between 0 and 1 is created and it is compared with the value of constant CR ($0 \leq CR \leq 1$), if the value of CR is less than the number created, the target vector will form that parameter of the child else the randomly chosen vector will form that parameter of the child When $CR = 1$ every parameter of the child vector will come from the randomly chosen vector On the other hand if $CR = 1$, all but one child vector parameter comes from the target vector To ensure that the child vector is not same as target vector the last parameter always comes from the randomly chosen vector

2.4 4 Selection

DE does not use any proportional selection, ranking for the selection Instead the cost of each trial

vector is compared with its parent target vector The vector with the lower cost is allowed to advance to the secondary array so as to form the population for the next generation In addition, if the trial vector wins, its cost is stored in $cost[\,]$

Some guidelines for choosing the values of population size, μ, the differential factor F and crossover constant have also been given by the authors as follows,

Ordinarily the population size ought to be about five to ten times of the number of parameters in a vector As for F, taking the value 0 5 initially is recommended and then increase the value of F if the premature convergence takes place A larger value of the constant CR is recommended

2.5 Family Competition Evolutionary Algorithm (FCEA)

The *FCEA* was proposed by (Yang, Lin and Kao, 2000) This algorithm is similar to evolution strategies Like evolution strategies, here also the mutation is the main search operator The main differences from the *ES* are

- Three different mutation operators are used here sequentially to produce the children population
- Two recombination operators have been applied to the solutions with associated probabilities
- Two selection techniques have been used to form the next generation

The basic *FCEA* is shown pictorially in Figure 2 14

The working of the *FCEA* is as follows,

N solutions are randomly generated as the initial population Then the *FCEA* enters the main evolutionary loop which consists of three similar procedures Each procedure is realized by doing recombinations, mutations, family competition and selection These three procedures mainly differs in the mutations used The mutations applied are decreasing Gaussian mutation, self adaptive Cauchy mutation and the self adaptive Gaussian mutation This procedure is termed as the FC adaptive procedure The procedure mainly employs four parameters, namely the parent population, (P with N solutions), mutation operator, (M), selection method, (S), and family competition length (L), to generate a new quasi-population The procedure to generate the new quasi-population is as follows

Each individual in the parent population sequentially becomes the *family father* With a probability p_c this *family father* and another randomly selected individual recombine to give an offspring Then the new offspring or the *family father* (if the crossover is not done) is mutated by the mutation operator specified For each *family father* this process is repeated L times to give L children From these L children points the best one is selected Since L children are created from the *family father* and a selection is performed, this is called a family competition strategy After the family competition there are total N

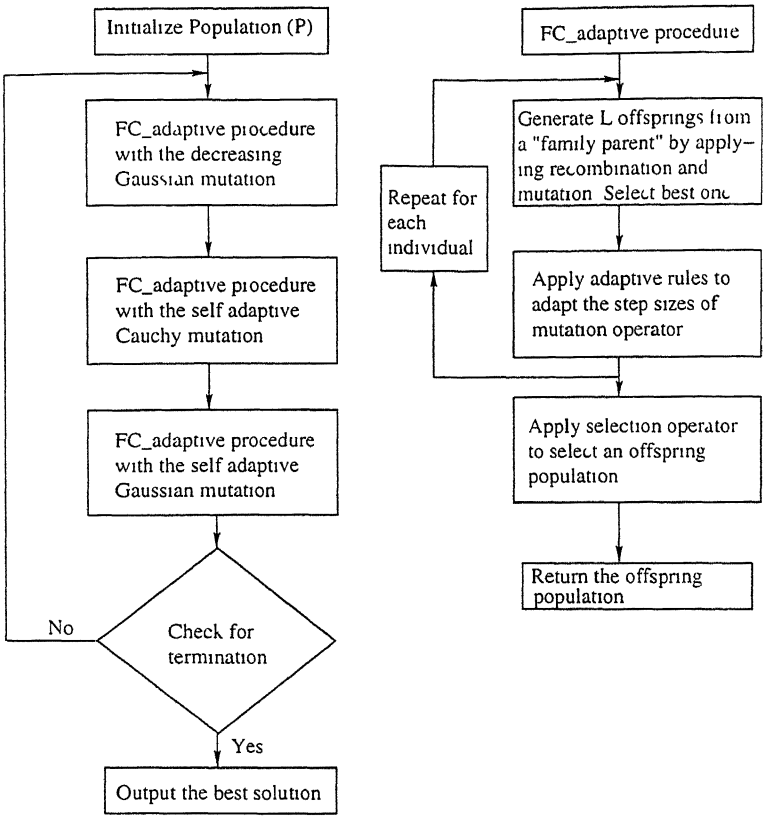


Figure 2 20 The family competition evolutionary algorithm and procedure used to do the mutation, recombination and selection

parents and N offsprings, now some selection scheme is used to generate the new quasi-population The selection scheme used depends on the mutation method used

In the following sections the various features of *FCEA* like representation, selection, recombination and mutation are described

2 5 1 Representation

The contents of the individual here are more or less similar to that in the evolution strategies Here an individual is represented as the set of four n-dimensional vectors $(x^i, \sigma^i, v^i, \psi^i)$, where n is the number of parameters and $i = 1,2 \quad N$ The vector x is the main variable vector to be optimized and σ v and ψ are the step size vectors of the decreasing mutation, self adaptive Gaussian mutation and self adaptive Cauchy mutation, respectively

2 5 2 Selection Procedures

As described earlier after family competition the selection is employed on the N parents and N offsprings to create N individuals for the new quasi-population The selection strategies used here are mainly "tournament selection" but with different forms These are as below,

Family Selection

In the family selection scheme the child is compared with its *family father* and the one out of these whichever is better is selected for the quasi-population

Population Selection

The population selection scheme selects N individual solutions from the population for the quasi-population Thus this selection is similar to the $(\mu + \mu)$ selection used in the traditional evolution strategies

The selection scheme used to form the quasi-population is dependent on the mutation operators In cases where Gaussian or Cauchy self-adaptive mutations are used in the FC adaptive procedures the Family Selection scheme is used while in the Population Selection scheme is used in case of decreasing Gaussian mutation In this case the population selection is used with a probability $(P - ps)$ and the family selection is used with probability $(1 - P_{ps})$ To reduce the ill effects of the greediness the initial value of P_{ps} is set to some small value e g 0 05 Later on when the mean step size of the self-adaptive Gaussian mutation is larger than that of the decreasing Gaussian mutation it is changed to some larger value e g 0 5

2 5 3 Recombination

The recombination in the proposed algorithms uses two operators namely modified discrete recombination and blend crossover (BLX-0 5) The intermediate recombination which is a special case of the BLX-0 5 is also used These operators are used with the probabilities 0 5, 0 25 and 0 25 respectively At each FC adaptive procedure only on of these operators is used to recombine with the probabilities mentioned before Here the recombination is same as that in the *ES* i e the recombination is done on problem variables as well as on the strategy parameters The recombination used are stated below

Modified Discrete Recombination

The original discrete recombination works by randomly choosing the genes from the two parents with equal parents Here this operator is modified so that the genes have higher probability to get selected from the "family father" The modified operator is as

$$x_j^c = \begin{cases} x_j^a \text{ with probability } 0\ 8 \\ x_j^b \text{ with probability } 0\ 2 \end{cases}$$

Here a b and c are family father, some randomly chosen vector and the child respectively $j \in (1 \quad n)$

BLX-0 5 and Intermediate Recombination

BLX-0 5 and Intermediate recombination work as already been described in the section 1 The operators recombine the problem parameter vectors as well as strategy parameters

2 5 4 Mutation

Mutations are the main search operators of *FCEA* After recombination, a mutation operator is applied to each "family father" or the new offspring generated by recombination In *FCEA* the mutation is performed independently on each vector element of the "family father" by adding a random value with expectation zero The mutation operators are described next,

Self-Adaptive Gaussian Mutation

The self-adaptive Gaussian mutation is accomplished by first mutating the step size v_j and then the variable x_j as

$$v_j^c = v_j^a exp\left[\tau' N(0,1) + \tau N_j(0,1)\right]$$

$$x_j^c = x_j^a + v_j^c N_j(0,1)$$

where $N(0,1)$ is the randomly generated number following normal distribution with mean 0 and standard deviation 1 $N_j(0,1)$ is sampled anew for each possible value of j The constants are $\tau = (\sqrt{2n})^{-1}$ and $\tau' = (\sqrt{2\sqrt{n}})^{-1}$

Self-Adaptive Cauchy Mutation

A random variable is said to have the Cauchy distribution if it has the following density function

$$f(x,t) = \frac{\frac{t}{\pi}}{t^2+x^2} \quad \text{for } -\infty < x < \infty$$

The mutation is performed as,

$$\psi_j^c = \psi_j^a exp\left[\tau' N(0,1) + \tau N_j(0,1)\right]$$

$$x_j^c = x_j^a + \psi_j^c C_j(1)$$

The different constants have already been defined ψ is the step size used in case of self-adaptive Cauchy's mutation

Decreasing Gaussian Mutation

The decreasing Gaussian mutation uses a fixed decreasing rate $\gamma = 0.95$ to control the step size. The mutation is done as follows,

$$\sigma^c = \gamma \sigma^a$$

$$x_j^c = x_j^a + \sigma^c N_j(0,1)$$

Chapter 3

Behaviour of Crossover Operators for Real-Coded GA

In this chapter the analysis of different crossover operators available for real-coded genetic algorithms is performed based on the *postulates* proposed by (Beyer and Deb, 1999) and the desirable properties of a crossover operator has been explored

3.1 Need

In genetic algorithms, two main operators are reproduction (selection) and variation (crossover) operators The function of the reproduction operator is to exploit the search space and to look for good regions in the search space The purpose of the variation operator is to use the parental population formed after the reproduction operation to create the children population In other words, the reproduction operator uses the information from the fitness space, whereas the variation operator handles the search space information Since an ideal variation operator may only process search space information of the selected population, the variation operator should not introduce any bias on the population level so the best it can do is to keep the mean of the parent population unchanged

There are several criteria for an efficient crossover operator There are some *postulates* suggested by Beyer and Deb (1999) for the efficient working of a crossover operator These *postulates* are described as below

3.1.1 Postulates on Population Mean

Postulate 1 *The expected population mean should remain unchanged under the application of a crossover operator*

Since the variation (crossover) operator, in general does not use the fitness information, there is no reason for it to shift the population mean in any direction It is the task of *reproduction operator* to provide the good regions to be exploited by the recombination operator In this way the best a crossover operator can do is to keep the mean of the population intact while increasing or decreasing the variance

of the population This also holds good for the standard binary GA recombination operators one-point n-point and uniform crossover in the genotype search space as well as in the resulting phenotype population

3 1 2 Postulates on Population Variance

The following postulates have been proposed for the different fitness landscapes,

Flat Fitness Landscapes

Postulate 2 *The expected population variance should exponentially increase with the generation number*

This is so as to leave the region of constant fitness as quickly as possible since there is no selective preference for these parts The best an algorithm can do under such circumstances is to randomly walk through the search space trying larger steps

Linear Fitness Landscapes(Hyper-plane)

Postulate 3 *The expected population variance should exponentially increase with generation number*

Linear fitness landscape divide the search space into two sub-spaces separated by a hyper-plane In Evolution Strategies, considering isotropic mutations, on average each second mutation (starting from a parental point) yields an improvement independent of the mutation strength used However the larger the mutations the larger the expected improvement Therefore, the mutation strength and the population variance, respectively should increase over the generations The same can be said for a recombination operator

Unimodal Fitness Landscapes

If the parental population brackets the optimum, a reduction in the variance of the population would serve the purpose but if it does not, a situation similar to that in case of linear fitness landscape arises and an increase in the variance would be beneficial In real world problems the latter case is more common because the location of the optima is not known apriori so an increase in the population variance would be better

Multi-modal Fitness Landscapes

The increase or decrease of the population variance should largely depend on the fitness landscape and the placement of the parent population It is argued that even in this case, a generic strategy would be to use a variation operator which increases the variance and then introduce adequate selection pressure by the reproduction operator to exploit the child population

In both of these cases it may be important to increase the variance after a certain period of shrinking So the algorithm should have the ability to increase the variance of the population

3.2 Analysis of Crossover Operators for Real-coded GAs in Flat Fitness Landscapes

In the following the analysis of a number of crossover operators in real-coded GAs is done for flat fitness landscapes Specifically, the mean and variance of the children population are derived from the known distribution of the parent population First the crossover operators which use two parents to generate two children are considered and then some multi-parental crossover operators are considered The theoretical results are compared with the simulations

Recombination Operators of a Special Type

In most of crossover operators in real-coded GAs, the recombination is done by picking two parents and creating two offsprings Since offsprings are created in pairs, we have $\mu/2$ pairs, here μ is the population size The paired individuals are numbered 1 and 2 and are represented by an index counter k running from 1 to $\mu/2$

The recombination operator proposed by Deb and Agrawal (1995) called simulated binary crossover (SBX) is defined as

$$y_{1,k} = \frac{1}{2}[(1-\beta_k)x_{1,k} + (1+\beta_k)x_{2,k}] \tag{3 1}$$

$$y_{2,k} = \frac{1}{2}[(1+\beta_k)x_{1,k} + (1-\beta_k)x_{2,k}] \tag{3 2}$$

Here $x_{1,k}$ and $x_{2,k}$ are independent samples from the parental population and β_k is a sample from a random number generator having the density

$$p(\beta) = \begin{cases} \frac{1}{2}(\eta+1)\beta^{\eta}, & \text{if } 0 \le \beta \le 1 \\ \frac{1}{2}(\eta+1)\frac{1}{\beta^{\eta+2}}, & \text{if } \beta \ge 1 \end{cases} \tag{3 3}$$

This distribution can easily be obtained from a uniform random number source by the transformation as

$$\beta(u) = \begin{cases} (2u)^{\frac{1}{\eta+1}}, & \text{if } u(0\ 1) \le 0\ 5 \\ [2(1-u)]^{-\frac{1}{\eta+1}}, & \text{if } u(0\ 1) > 0\ 5 \end{cases} \tag{3 4}$$

Beyer and Deb (1999) formulated the mean and variance of the population as follows,

Definition of Population Mean

The population mean is defined as

$$\langle y \rangle = \frac{1}{\mu} \sum_{m=1}^{\mu} y_m \tag{3 5}$$

Here the subscript m is the individual counter running from 1 to population size μ and $\langle y \rangle$ is the average or mean of the offspring population Since the crossover operation produces two offsprings, rearranging above equation to

$$\langle y \rangle = \frac{1}{\mu} \sum_{k=1}^{\mu/2} (y_{1,k} + y_{2,k}) \tag{3 6}$$

Definition of Population Variance

The population variance of the offspring is defined as,

$$\mathrm{Var}\{y\} = \frac{1}{\mu} \sum_{m=1}^{\mu} (y_m - \langle y \rangle)^2 \tag{3 7}$$

The *expected* value of above expression will be,

$$\sigma^2 [y] = E [Var\{y\}] = \overline{Var\{y\}} \tag{3 8}$$

it depends on the *expected* variance of the parental population, denoted by $\sigma^2 [x]$ which is given by the following equation,

$$\sigma^2 [x] = E \left[(x - \overline{x})^2 \right] = \overline{x^2} - \overline{x}^2 \tag{3 9}$$

Calculation of the population average $\langle y \rangle$

Using eqns (3 1) and (3 2) in (3 6), we get

$$\langle y \rangle = \frac{1}{\mu} \sum_{k=1}^{\mu/2} (x_{1,k} + x_{2,k}) = \frac{1}{\mu} \sum_{m=1}^{\mu} x_m \tag{3 10}$$

Thus crossover operator represented by eqns (3 1) and (3 2) preserves the mean of the parental population As stated earlier, this is a desirable property of the crossover operators by Postulate 1

Calculation of population variance

From eqn (3 7) we can alternatively write,

$$Var\{y\} = \frac{1}{\mu}\sum_{m=1}^{\mu}(y_m - \langle y\rangle)^2 = \frac{1}{\mu}\sum_{k=1}^{\mu/2}\left[(y_{1,k} - \langle y\rangle)^2 + (y_{2,k} - \langle y\rangle)^2\right] \tag{3 11}$$

As is clear taking (3 1),(3 2) and (3 10) into account, the Var{y} consists of a sum over $x_{i,k}x_{j,l}$ products Since the expected value of Var{y} is to be determined, we have to determine the expected value of $x_{i,k}x_{j,l}$ terms Since the sampling process copies the individuals at random from the parental pool to the crossover operator, all $x_{i,k}$ are independent and identically distributed Therefore we have

$$\overline{x_{i,k}x_{j,l}} = \begin{cases} \overline{x^2}, & \text{if } i = j \wedge k = l \\ \overline{x}^2, & \text{otherwise} \end{cases} \tag{3 12}$$

where $\overline{x}$ and $\overline{x^2}$ are the first and second moments respectively, of the parental population distribution In order to calculate the expectation of (3 11), the square brackets in (3 11) must be rearranged in such a way that the products $x_{i,k}x_{j,l}$ are separated from the rest This can be done as follows,
Beyer and Deb (1999) found the following variance evolution equation for above type of crossover operators as follows

$$\sigma^{2(g+1)} = \frac{1}{2}\left(1 - \frac{2}{\mu} + \overline{\beta^2}\right)\sigma^{2(g)} \tag{3 13}$$

By recurrence, the equation of expected variance dynamics can be written as,

$$\sigma^2[y]^{(g)} = \sigma^2[y]^{(0)}\left(\frac{1}{2} - \frac{1}{\mu} + \frac{\overline{\beta^2}}{2}\right)^g \tag{3 14}$$

for the derivation of the above equation, see Appendix A
It is clear from above equation that there is an exponential change in the population variance with generation number g

3 2 1 Determination of Variance Evolution Equation for Different Crossover Operator

In this section value of $\overline{\beta^2}$ for different crossover operators will be evaluated and the variance evolution equation will be obtained Beyer and Deb (1999) obtained the variance evolution equation for SBX BLX and FR crossover operators which is also presented in the following and the evolution equation for other operators is also determined The evolution equation (3 14) contains the term $\overline{\beta^2}$ which depends on the random number distribution used Its value is given as

$$\overline{\beta^2} = \int \beta^2 p(\beta)\, d\beta \tag{3 15}$$

The variance evolution equation for different crossover operators is obtained as below,

SBX-η

In case of SBX the standard way to generate β is by transformation from the uniform variate u(0,1) Thus we can rewrite eqn (3 15) as

$$\overline{\beta^2} = \int_{u=0}^{u=1} (\beta(u))^2 \, du \tag{3 16}$$

Using eqn (3 4) and eqn (3 16), we arrive at

$$\overline{\beta^2} = \int_{u=0}^{u=\frac{1}{2}} (2u)^{\frac{2}{\eta+1}} \, du + \int_{u=\frac{1}{2}}^{u=1} [2(1-u)]^{-\frac{2}{\eta+1}} \, du \tag{3 17}$$

Making substitution $s = 2u$ in the first integral and $t = 2(1-u)$ in the second integral we get following equation for $\eta \neq 1$

$$\overline{\beta^2} = \frac{1}{2}\left[\frac{1}{1+\frac{2}{\eta+1}} s^{\left(1+\frac{2}{\eta+1}\right)}\right]_0^1 + \frac{1}{2}\left[\frac{1}{1-\frac{2}{\eta+1}} t^{\left(1-\frac{2}{\eta+1}\right)}\right]_0^1 \tag{3 18}$$

which on substitution gives

$$\overline{\beta^2} = \frac{1}{2}\left[\frac{\eta+1}{\eta+3} + \frac{\eta+1}{\eta-1}\right] \tag{3 19}$$

simplifying the equation we get,

$$\overline{\beta^2} = \frac{(\eta+1)^2}{(\eta+3)(\eta-1)} \tag{3 20}$$

Using eqn (3 20) in eqn (3 14), we have

$$\sigma^2[y]^{(g)} = \sigma^2[y]^{(0)} \left(1 + \frac{2}{(\eta+3)(\eta-1)} - \frac{1}{\mu}\right)^g \tag{3 21}$$

Above equation gives the variance dynamics in case of SBX crossover operator It holds good for $\eta > 1$ As is clear from the equation the expected population variance depends on the value of η parameter in case of SBX for a given population size μ From the above expression we can easily determine the value of parameter η for which the population variance will increase for a given population size

For variance to increase, the terms within large parenthesis should be greater than one Doing so we get

$$\mu > \frac{1}{2}(\eta+3)(\eta-1) \tag{3 22}$$

Thus for a particular η, the population size μ should be greater than the above value for expected population variance to increase

In case of BLX-α we create two offsprings as given in section 2 1 1, comparison with eqns (3 1) and (3 2) yields $\beta = 2\xi - 1$ Here ξ is a uniformly generated random number in the range $[-\alpha, 1+\alpha]$ so the probability distribution will be $p(\xi) = \frac{1}{1+2\alpha}$ From eqn (3 15) we have,

$$\overline{\beta^2} = \frac{1}{1+2\alpha}\int_{\xi=-\alpha}^{\xi=1+\alpha} (2\xi - 1)^2 \, d\xi$$

$$= \frac{1}{1+2\alpha}\int_{\xi=-\alpha}^{\xi=1+\alpha} \left(4\xi^2 - 4\xi + 1\right) d\xi$$

$$= \frac{1}{1+2\alpha}\left[\frac{4}{3}\xi^3 - 2\xi^2 + \xi\right]_{-\alpha}^{1+\alpha}$$

which on simplification yields

$$\overline{\beta^2} = \frac{1}{3}(2\alpha + 1)^2 \tag{3 23}$$

Using above equation in eqn (3 14), we get the variance evolution equation for BLX as

$$\sigma^2[y]^{(g)} = \sigma^2[y]^{(0)}\left(1 + \frac{1}{3}\left(2\alpha^2 + 2\alpha - 1\right) - \frac{1}{\mu}\right)^g \tag{3 24}$$

Again it is clear from above equation that the population variance increases or decreases, it depends on the population size for a given value of parameter α which means there will be certain value of population size for a given value of α over which the population variance will increase and at this particular value of population size the convergent and divergent will be balanced

This minimum value of population size which gives diverging population can be determined by putting terms inside brackets greater than 1 of eqn (3 24) This gives us

$$\mu > \frac{3}{2\alpha + 2\alpha^2 - 1} \tag{3 25}$$

FR-d

In fuzzy recombination operator suggested by Voigt et al (1995) two offsprings are created using a triangular probability distribution as described in section 2 1 1, the eqn (3 15) can be written in this case as

$$\overline{\beta^2} = \int_{\zeta=-1}^{\zeta=1} p(\zeta) \, d\zeta$$

Here ζ is a randomly generated number following a triangular distribution and the probability distribution is given as

के द्रीय पुस्तकालय
[illegible] है० कानपुर

$$p(\zeta) = \begin{cases} \zeta + 1 & \text{for } -1 \leq \zeta \leq 0 \\ 1 - \zeta & \text{for } 0 \leq \zeta \leq 1 \end{cases} \tag{3 26}$$

Comparison of equations for offsprings in case of FR as given in section 2 1 1 with eqns (3 1) and (3 2) yields the value of β for FR to be $1 + 2\zeta d$ Using this value β and eqn (3 26) in (3 15) we have

$$\overline{\beta^2} = \int_{\zeta=-1}^{\zeta=0} (1 + 2\zeta d)^2 (\zeta + 1)\, d\zeta + \int_{\zeta=0}^{\zeta=1} (1 + 2\zeta d)^2 (1 - \zeta)\, d\zeta$$

which on integration gives

$$\overline{\beta^2} = 1 + \frac{2}{3} d^2 \tag{3 27}$$

substituting this value of $\overline{\beta^2}$ in equation (3 14), we get variance dynamics for the FR operator as

$$\sigma^2 [y]^{(g)} = \sigma^2 [y]^{(0)} \left(1 + \frac{1}{3} d^2 - \frac{1}{\mu}\right)^g \tag{3 28}$$

In this case too, by the same argument we can obtain the value of population size for a given value of d for which the expected population variance will increase This will give us following value of population size

$$\mu > \frac{3}{d^2} \tag{3 29}$$

Flat Crossover

This crossover operator is same as the BLX-0 0, thus we can write the variance evolution equation for it by putting $\alpha = 0$ in eqn (3 24) as

$$\sigma^2 [y]^{(g)} = \sigma^2 [y]^{(0)} \left(\frac{2}{3} - \frac{1}{\mu}\right)^g \tag{3 30}$$

It is clear from above equation that this operator reduces the variance of the population independent of the population size

Arithmetical Crossover

This operator produces offsprings as described in section 2 1 1, the parameter λ is used to create children Comparison of expressions for offsprings with (3 1) and (3 2) gives value of β as $2\lambda - 1$ which is a constant, therefore the probability density function for the parameter β would be unity Now using (3 15) we have

$$\overline{\beta^2} = (2\lambda - 1)^2 \tag{3 31}$$

Substituting in (3 14), we get

$$\sigma^2 [y]^{(g)} = \sigma^2 [y]^{(0)} \left(1 + 2\lambda(\lambda - 1) - \frac{1}{\mu}\right)^g \tag{3 32}$$

From the above equation, it is obvious that any value of $\lambda \leq 1$ will reduce the variance of the population irrespective of the population size μ

For values of $\lambda > 1$, the variance will increase when the population size will follow the following equation

$$\mu > \frac{1}{2\lambda(\lambda - 1)} \tag{3 33}$$

Extended Line and Extended Intermediate Crossover

Both these crossover operators are same as BLX-0 25, the variance evolution equation can be written by substituting $\alpha = 0\ 25$ in (3 24) It will be as below,

$$\sigma^2 [y]^{(g)} = \sigma^2 [y]^{(0)} \left(\frac{7}{8} - \frac{1}{\mu}\right)^g \tag{3 34}$$

As one can see that the population variance will decrease irrespective of the population size chosen

The analysis for the evolution of variance has been done above for the operators which generate two children from two parents In the following other crossover operators like UNDX and SPX will be considered First UNDX has been considered

Unimodal Normal Distribution Crossover:

The UNDX generated offsprings using three parents and two normal distributions, one along the line connecting first two parents and the other one along the line which is normal to the line connecting first two parents The children points are generated as follows

$$\mathbf{y^1} = \mathbf{x_p} + \xi \mathbf{d} + \sum_{j=1}^{n-1} \eta_j D \mathbf{e_k}$$

$$\mathbf{y^2} = \mathbf{x_p} - \xi \mathbf{d} - \sum_{j=1}^{n-1} \eta_j D \mathbf{e_k}$$

Previous Work:

Kita et al (1998) have done the analysis for the UNDX operator for mean preservation and covariance preservation in *infinite* population Their findings are stated in the following,

Mean From the equation of the child solution above in UNDX operator taking expected value of the expression yields

$$\begin{aligned} < \mathbf{y}^1 > &= \left\langle \mathbf{x_p} + \xi \mathbf{d} + \textstyle\sum_{j=1}^{n-1} \eta_j D \mathbf{e_k} \right\rangle \\ &= \langle \mathbf{x_p} \rangle + \langle \xi \rangle \langle \mathbf{d} \rangle + \textstyle\sum_{j=1}^{n-1} \langle \eta \rangle \langle \mathbf{e_k} D \rangle \\ &= < \mathbf{x_p} > \\ &= \tfrac{1}{2}(< \mathbf{x}^1 > + < \mathbf{x}^2 >) \\ < \mathbf{y}^1 > &= \overline{\mathbf{x}} \end{aligned}$$

It shows that the UNDX operator preserves mean for *infinite* population

Covariance For the covariance analysis, they considered only the parallel component of the crossover operator The equation they got is as below,

$$\gamma_{ij}^{y} = \gamma_{ij}^{x} \left(\frac{1}{2} + 2\sigma_{\xi}^{2} \right) \tag{3 35}$$

It can be seen from the above equation that the covariance matrix of the population can expand or contract depending upon the value of the constant σ_ξ

Analysis of UNDX operator for finite population·

In the following the UNDX will be checked for *postulate 1* i e mean preservation and *postulate 2* i e variance dynamics" for finite population First the mean is considered

Mean The UNDX creates points as given by above two equations, substituting these equations in the eqn (3 6) we get

$$< \mathbf{y} >= \frac{1}{\mu} \sum_{k=0}^{k=\mu/2} (\mathbf{y_k^1} + \mathbf{y_k^2})$$

$$< \mathbf{y} >= \frac{1}{\mu} \sum_{k=0}^{k=\mu/2} 2\mathbf{x}_{\mathbf{p},\mathbf{k}}$$

since $\mathbf{x}_{\mathbf{p},\mathbf{k}} = \frac{\mathbf{x_k^1} + \mathbf{x_k^2}}{2}$, we have

$$< \mathbf{y} >= \frac{1}{\mu} \sum_{k=0}^{k=\mu/2} (\mathbf{x_k^1} + \mathbf{x_k^2})$$

which can be written as

$$< \mathbf{y} >= \frac{1}{\mu} \sum_{i=0}^{i=\mu} \mathbf{x}_i =< \mathbf{x} >$$

where $< \mathbf{x} >$ is the mean of the parental population, therefore the mean of the parents is preserved by the UNDX operator So it satisfies *postulate 1* Here $\mathbf{y}$ is the n-dimensional vector

Variance Analysis of the UNDX operator The term D in the equations of the children points as obtained by the UNDX is the normal distance of the third parent from the line connecting first two parents D is given by the following expression

$$D^2 = |\mathbf{x}^3 - \mathbf{x}^1|^2 \left(1 - \left(\frac{(\mathbf{x}^3 - \mathbf{x}^1)^T(\mathbf{x}^2 - \mathbf{x}^1)}{|\mathbf{x}^3 - \mathbf{x}^1||\mathbf{x}^2 - \mathbf{x}^1|}\right)^2\right) \tag{3 36}$$

An analysis similar to that has been performed for the operators described above would involve in this case the determination of the expectation of the quantity $\left(\sum_{j=1}^{j=n-1}\eta_j De_j\right)^2$ which would further have the parents in the denominator too This is difficult to evaluate because the distribution of the parental population is unknown, however a simplified analysis can be done by considering the parallel component only i e considering only the first two terms in the equations for the generations of the children solutions Considering the parallel component only the children points as generated by the UNDX operator will be given as,

$$y_i^1 = x_{p,i} + \xi d_i \tag{3 37}$$

$$y_i^2 = x_{p,i} - \xi d_i \tag{3 38}$$

It has the same form as given by equations (3 1) and (3 2) Above equations are for the i^{th} component A comparison with the eqns (3 1) and (3 2) yields value of parameter β in this case as

$$\beta = 2\xi$$

here ξ is the normally distributed random number following $N(0, \sigma_\xi^2)$ To get the variance evolution equation in this case, we need to calculate the quantity $\overline{\beta^2}$ In equation (3 15), the probability distribution for β is given by

$$p(\beta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\xi^2}{2\sigma^2}} \tag{3 39}$$

Using above eqn and $\beta = 2\xi$ in equation (3 15), we get

$$\overline{\beta^2} = \int_{-\infty}^{\infty} (2\xi)^2 \frac{1}{\sigma_\xi\sqrt{2\pi}} e^{-\frac{\xi^2}{2\sigma^2}} d\xi \tag{3 40}$$

$$\overline{\beta^2} = 4\int_{-\infty}^{\infty} \xi^2 \frac{1}{\sigma_\xi\sqrt{2\pi}} e^{-\frac{\xi^2}{2\sigma^2}} d\xi \tag{3 41}$$

substituting $\frac{\xi}{\sigma\sqrt{2}} = t$, we get

$$\overline{\beta^2} = \frac{4\sigma^2}{\sqrt{\pi}} \int_{-\infty}^{\infty} t \;\; 2t \;\; e^{-t^2} dt \tag{3 42}$$

$$\overline{\beta^2} = \frac{4\sigma^2}{\sqrt{\pi}} \left[\left(-te^{-t^2}\right)_{-\infty}^{\infty} + \int_{-\infty}^{\infty} e^{-t^2} dt\right] \tag{3 43}$$

Since $\lim_{t\to\infty} te^{-t^2} = 0$ and $\int_{-\infty}^{\infty} e^{-t^2}dt = \sqrt{\pi}$, substituting these above, we have

$$\overline{\beta^2} = 4\sigma^2 \tag{3 44}$$

Using this value of $\overline{\beta^2}$ in eqn (3 14), we get the variance evolution equation for the UNDX operator as

$$\sigma^2[y]^{(g)} = \sigma^2[y]^{(0)}\left[1 + \frac{4\sigma^2 - 1}{2} - \frac{1}{\mu}\right]^g \tag{3 45}$$

Here $\sigma = \sigma_\xi$ From the above evolution equation, for variance to increase, the terms in the bracket need to be greater than 1 Doing so yields,

$$\mu > \frac{2}{4\sigma_\xi^2 - 1} \tag{3 46}$$

Thus for a given value of σ_ξ the population size should be greater than above value Note that this equation is for the UNDX operator with only parallel component but it can give an idea on how much population size should be chosen to get the desired behaviour of the crossover operation

Remarks

Comparing the equations (3 35) and (3 45), we see that the both equations are same except in equation (3 45) the population size term also comes It means that the variance evolution of population depends on the population size also The minimum population size which increases the population variance is given by eqn (3 45) Kita et al (1998) suggested the value of the term σ_ξ^2 to be 0 25 to keep the covariance of the population same but as can be seen from equation (3 45) that for this value of σ_ξ^2 the variance of the population will decrease by a factor of $\frac{1}{\mu}$ over the generations The validity of the equation (3 45) is also demonstrated later in this Chapter

Simplex Crossover. The Simplex Crossover generates one offspring by recombining $m+1$ parents where m is number of decision variables The procedure of creating the offspring is restated in the following

1 Choose $m+1$ parents from the parental population randomly Let the parents be $\mathbf{x^0}$ $\mathbf{x^2}$, $\mathbf{x^m}$

2 Calculate the centre of mass of the parents chosen above as,

$$\mathbf{x_p} = \frac{1}{m+1}\sum_{j=0}^{m}\mathbf{x^j} \tag{3 47}$$

3 Generate r_k, which is random number given by,

$$\mathbf{r_k} = (u(0,1))^{\frac{1}{k+1}} \text{ for } k = 0, \quad , m-1 \tag{3 48}$$

Here $u(0,1)$ is uniformly generated random number between 0 and 1

4 Calculate $\mathbf{X_k}$ and $\mathbf{C_k}$ as below,

$$\mathbf{X_k} = \mathbf{x_p} + \epsilon \left(\mathbf{x^k} - \mathbf{x_p} \right) \tag{3 49}$$

$$\mathbf{C_k} = \begin{cases} 0 & \text{for } k = 0 \\ \mathbf{r_{k-1}}(\mathbf{X_{k-1}} - \mathbf{X_k} + \mathbf{C_{k-1}}) & \text{for } k = 1, \quad , m \end{cases} \tag{3 50}$$

where ϵ is the expansion rate

5 An offspring is given by

$$C = \mathbf{X_m} + \mathbf{C_m} \tag{3 51}$$

Previous Work

Higuchi Tsutsui and Yamamura (2000) did the analysis of the SPX operator for *infinite* population case They determined the mean and covariance of the population for *infinite* population which are described in the following

Mean Considering the mean of the parental population as $< x >$ This and (3 49) yield $< X_{k-1} - X_k >= 0$ Using this and (3 50), we get $< C_k >=< r_{k-1} >< C_{k-1} >= 0$ by induction Using this and (3 51) we get $< C >=< C_m > + < X_m >=< x >$

Thus the mean of the parental population is preserved by the SPX operator for *infinite* population

Covariance Let $(m + 1)$ denote the number of parents The covariance matrix of the distribution of the offsprings given by SPX is given by,

$$\gamma_{ij}^{c} = \frac{1}{m+1} \left(1 + \epsilon^2 \frac{m}{m+2} \right) \gamma_{ij}^{x} \tag{3 52}$$

from above equation it can be seen that the covariance matrix can expand or contract depending on the value of characteristic parameter ϵ and number of parents taking part in recombination, $m + 1$

Analysis of SPX operator for finite population.

In this section the analysis of the SPX operator will be done as according to the formulation done by Beyer and Deb (1999) Since the SPX operator uses more than two parents, the analysis has been done considering different cases of recombination involving two parents, three parents and a generalized equation for M parents has been obtained In the following these cases will be considered and mean and variance characteristics of the children population will be checked In the analysis all parents, expanded parents factors C and the child are vectors and the calculations will hold for all the variables

Two Parent Case

In case of two parents, let x_0 and x_1 be the parents The centre of mass of the parents is given as

$$G = \frac{x_0 + x_1}{2} \tag{3 53}$$

The random number $r_0 = u$ where u is a uniformly generated random number between 0 and 1 The parents are expanded to X_0 and X_1 by a factor of expansion rate ϵ, with respect to the centre of mass as

$$X_0 = \frac{1+\epsilon}{2}x_0 + \frac{1-\epsilon}{2}x_1 \tag{3 54}$$

$$X_1 = \frac{1-\epsilon}{2}x_0 + \frac{1+\epsilon}{2}x_1 \tag{3 55}$$

The factors C_0 and C_1 are given as

$$C_0 = 0$$

$$C_1 = r_0\epsilon\,(x_0 - x_1)$$

The offspring will be created as $y = X_1 + C_1$ and would be

$$y = \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_0 + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_1 \tag{3 56}$$

Mean of the children population The mean of the children population as generated by the SPX will be

$$< y >= \frac{1}{\mu}\sum_{l=1}^{l=\mu} y_l \tag{3 57}$$

Substituting for y_m from the eqn (1 54) into the above equation, we get

$$< y >= \frac{1}{\mu}\sum_{m=1}^{m=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right] \tag{3 58}$$

As is clear from the above equation that the SPX does not preserve the mean of the parental population in case of the finite population instead it depends on the random number r_0 However if we take the population infinite it preserves the mean as is shown below,

$$< y >= \left\langle\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_0 + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_1\right\rangle$$

since $< r_0 >=< u >= \frac{1}{2}$ and $< x_0 >=< x_1 >= \overline{x}$, we have

$$= \left(\frac{1-\epsilon}{2} + \frac{\epsilon}{2}\right)\overline{x} + \left(\frac{1+\epsilon}{2} - \frac{\epsilon}{2}\right)\overline{x}$$

which will finally give $< \mathbf{y} > = < \mathbf{x} > = \overline{\mathbf{x}}$ Thus SPX preserves the mean of the parental population in case of infinite population but for finite population it does not

Variance of the children population From equation (3 8), we have variance of the children population as

$$\sigma^2[y] = E[Var\{y\}] = \overline{Var\{y\}} \tag{3 59}$$

here $Var\{y\}$ is defined as

$$Var\{y\} = \frac{1}{\mu}\sum_{m=1}^{m=\mu}(y_m - < y >)^2 \tag{3 60}$$

So to determine the expected population variance of the children population, we need to calculate the expectation of the quantity $(y_m - < y >)^2$ Here too the determination of this quantity will involve the products $x_{i,k}x_{j,l}$ and the expectation of these products is defined as,

$$\overline{x_{i,k}x_{j,l}} = \begin{cases} \overline{x^2}, & \text{if } i = j \wedge k = l \\ \overline{x}^2, & \text{otherwise} \end{cases} \tag{3 61}$$

Now considering the k^{th} sum terms, we have

$$y_k - < y >= \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,k} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,k} - \frac{1}{\mu}\sum_{l=0}^{l=\mu}\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}$$

$$= \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{0,k} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{1,k}$$

$$-\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right] \tag{3 62}$$

Squaring the above equation gives,

$$(y_k - < y >)^2 = \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)^2\left(1 - \frac{1}{\mu}\right)^2 x_{0,k}^2 + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)^2\left(1 - \frac{1}{\mu}\right)^2 x_{1,k}^2$$

$$+\frac{1}{\mu^2}\left[\sum_{l\neq k}^{l=\mu}\left(\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right)\right]^2 + 2\left[\frac{1-\epsilon^2}{4} + r_0\epsilon^2 - r_0^2\epsilon^2\right]\left(1 - \frac{1}{\mu}\right)^2 x_{0,k}x_{1,l}$$

$$-\frac{2}{\mu}\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{0,k}\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right]$$

$$-\frac{2}{\mu}\left(\frac{1+\epsilon}{2} - r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{1,k}\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right]$$

After substituting in (3 60) taking (3 61) into account, we get

$$\sigma^2[y] = \frac{1}{\mu}\sum_{m=1}^{m=\mu}E(y_m - < y >)^2$$

Variance of the children population From equation (3 8), we have variance of the children population as

$$\sigma^2[y] = E[Var\{y\}] = \overline{Var\{y\}} \tag{3 59}$$

here Var{y} is defined as

$$Var\{y\} = \frac{1}{\mu}\sum_{m=1}^{m=\mu}(y_m - < y >)^2 \tag{3 60}$$

So to determine the expected population variance of the children population, we need to calculate the expectation of the quantity $(y_m - < y >)^2$ Here too the determination of this quantity will involve the products $x_{i,k}x_{j,l}$ and the expectation of these products is defined as,

$$\overline{x_{i,k}x_{j,l}} = \begin{cases} \overline{x^2}, & \text{if } i = j \wedge k = l \\ \overline{x}^2, & \text{otherwise} \end{cases} \tag{3 61}$$

Now considering the k^{th} sum terms, we have

$$y_l - < y >= \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,k} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,k} - \frac{1}{\mu}\sum_{l=0}^{l=\mu}\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}$$

$$= \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{0,k} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{1,k}$$

$$-\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right] \tag{3 62}$$

Squaring the above equation gives,

$$(y_k - < y >)^2 = \left(\frac{1-\epsilon}{2} + r_0\epsilon\right)^2\left(1 - \frac{1}{\mu}\right)^2 x_{0,k}^2 + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)^2\left(1 - \frac{1}{\mu}\right)^2 x_{1,k}^2$$

$$+\frac{1}{\mu^2}\left[\sum_{l\neq k}^{l=\mu}\left(\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right)\right]^2 + 2\left[\frac{1-\epsilon^2}{4} + r_0\epsilon^2 - r_0^2\epsilon^2\right]\left(1 - \frac{1}{\mu}\right)^2 x_{0,k}x_{1,k}$$

$$-\frac{2}{\mu}\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{0,k}\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right]$$

$$-\frac{2}{\mu}\left(\frac{1+\epsilon}{2} - r_0\epsilon\right)\left(1 - \frac{1}{\mu}\right)x_{1,k}\frac{1}{\mu}\sum_{l\neq k}^{l=\mu}\left[\left(\frac{1-\epsilon}{2} + r_0\epsilon\right)x_{0,l} + \left(\frac{1+\epsilon}{2} - r_0\epsilon\right)x_{1,l}\right]$$

After substituting in (3 60) taking (3 61) into account, we get

$$\sigma^2[y] = \frac{1}{\mu}\sum_{m=1}^{m=\mu}E(y_m - < y >)^2$$

$$\sigma^2[y] = \frac{1}{\mu}\sum_{m=1}^{m=\mu}\left[\frac{1+\epsilon^2}{2} + 2\overline{r_0^2}\epsilon^2 - 2\overline{r_0}\epsilon^2 - \frac{1}{\mu}\left(\frac{1+\epsilon^2}{2} + 2\overline{r_0^2}\epsilon^2 - 2\overline{r_0}\epsilon^2\right)\right]\left(\overline{x^2} - \overline{x}^2\right)$$

Since $\overline{r_0} = \frac{1}{2}$ and $\overline{r_0^2} = \frac{1}{3}$, we have from the above equation after substitution,

$$\sigma^{2(g+1)}[y] = \frac{1}{2}\left[1 + \frac{\epsilon^2}{3} - \frac{1}{\mu}\left(1 + \frac{\epsilon^2}{3}\right)\right]\sigma^{2(g)}[y] \tag{3 63}$$

This gives the variance dynamics for the SPX crossover operator in case of 2 parents

Three parent case

In this case, let x_0, x_1 and x_2 be the parents, so the centre of mass of the parents is given as

$$G = \frac{1}{3}(x_0 + x_1 + x_2) \tag{3 64}$$

The random numbers r_0 and r_1 will be $r_0 = u(0\ 1)$ and $r_1 = (u(0,1))^{\frac{1}{2}}$ Here u(0,1) is a uniformly generated random number between 0 and 1

The vertices of the extended simplex will be

$$X_0 = \frac{1+2\epsilon}{3}x_0 + \frac{1-\epsilon}{3}x_1 + \frac{1-\epsilon}{3}x_2 \tag{3 65}$$

$$X_1 = \frac{1-\epsilon}{3}x_0 + \frac{1+2\epsilon}{3}x_1 + \frac{1-\epsilon}{3}x_2 \tag{3 66}$$

$$X_2 = \frac{1-\epsilon}{3}x_0 + \frac{1-\epsilon}{3}x_1 + \frac{1+2\epsilon}{3}x_2 \tag{3 67}$$

The factors C_0, C_1 and C_2 are given as

$$C_0 = 0$$

$$C_1 = r_0\epsilon(x_0 - x_1)$$

$$C_2 = r_0 r_1 \epsilon x_0 + (r_1\epsilon - r_0 r_1 \epsilon)x_1 - r_1\epsilon x_2$$

The offspring will be,

$$y = X_2 + C_2 = \left(\frac{1-\epsilon}{3} + r_0 r_1 \epsilon\right)x_0 + \left(\frac{1-\epsilon}{3} + r_1\epsilon - r_0 r_1 \epsilon\right)x_1 + \left(\frac{1+2\epsilon}{3} - r_1\epsilon\right)x_2 \tag{3 68}$$

Remarks

From the above equation it is clear that the variance of the population generated by the SPX operator depends on the characteristic parameter ϵ, number of parents taking part for the crossover and on the population size too

For the variance to increase as by *postulate 2*, the quantity in the brackets on the right side of above equation should be greater than M This gives us the minimum population size for a given value of parameter ϵ and M as follows,

$$\mu > \frac{1 + \frac{M-1}{M+1}\epsilon^2}{1 + \frac{M-1}{M+1}\epsilon^2 - M} \tag{3 73}$$

If we compare the equations (3 52) and (3 71), we find that both equation are same up to two terms In case of finite population the we see that the variance evolution also depends on the population size Later in this Chapter it is validated with the simulations The investigators of SPX have argued that to keep the covariance of the population same the characteristic parameter should follow the relation

$$\epsilon = \sqrt{M+1}$$

It is clear from equation (3 71) that this value of ϵ will always reduce the variance and in turn the covariance of the population no matter how much population size μ we take For the variance of the population to increase the value of population size should follow equation (3 73)

One more thing to note here is that for a given value of the number of parents the value of parameter ϵ should be greater than $(M+1)$ for variance to increase If $\epsilon < M+1$, the variance of the population will always decrease independent of the population size

3 2.2 Comparison with Simulation Results

The predictive quality of the evolution equation for different crossover operator can be evaluated for different operators by making GA runs on a flat fitness landscape Here theoretical prediction are being compared with the simulation results for SBX-η, BLX-α, FR-d, UNDX-σ_ξ and SPX-M-ϵ

Figure 3 1 compares the formula (3 21) of the population variance dynamics (line represented with a + and dotted line respectively) with the simulation using SBX operator Figure 3 2 compares the formula (3 21) of the population variance dynamics (line represented with a + and dotted line respectively) with the simulation using SBX operator

Figure 3 3 and Figure 3 4 compares the results of the GA runs for BLX-0 0 and BLX-0 5 respectively with the prediction given by equation (3 24) Figure 3 5 and Figure 3 6 compares the simulations of the RGA using FR-0 5 and FR-0 7 operators with the evolution equation given by (3 28)

Figure 3 7 shows the variance dynamics in case of UNDX without perpendicular component as given by equation (e = 60 with $\sigma_\xi = 0\ 75$ and for popsize=6 with $\sigma_\xi = 0\ 55$ Figure 3 8 shows

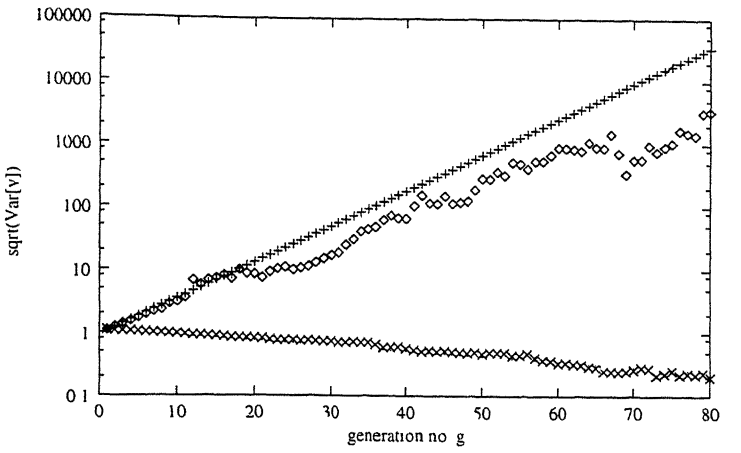


Figure 3 1 Evolution of the square root of the expected population variance for the SBX operator in the flat fitness landscape Population size=10 with $\eta = 2$ (upper line, displayed by +, simulations displayed by $\diamond$) and $\eta = 5$ (lower, dotted line, simulations displayed by $\times$) The population has been randomly initialized with variance 1 The simulation results have been obtained by averaging over 3000 independent runs

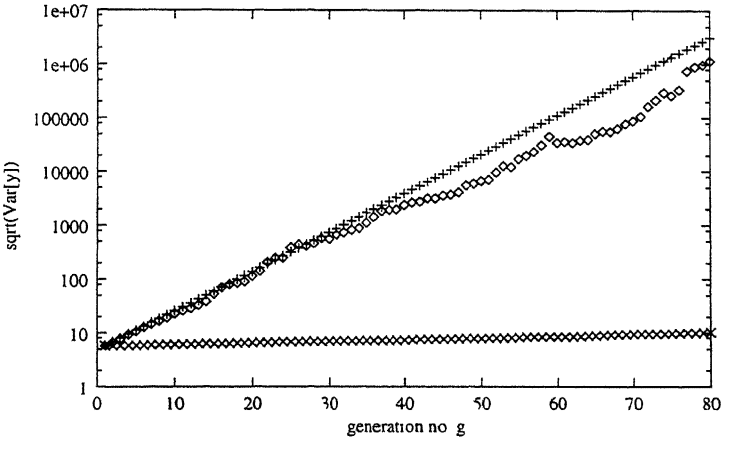


Figure 3 2 Evolution of the square root of the expected population variance for the SBX operator in the flat fitness landscape Population size=300 with $\eta = 2$ (upper line, displayed by +, simulations displayed by $\diamond$) and $\eta = 10$ (lower, dotted line, simulations displayed by $\times$) The population has been randomly initialized with variance 6 The simulation results have been obtained by averaging over 100 independent runs

the variance evolution as given by equation (3 45) for popsize = 60 with $\sigma_\xi = 0\ 50$ and for popsize = 10 with $\sigma_\xi = 0\ 35$

Figure 3 9 and Figure 3 10 show the evolution of variance according to the formula (3 72) in case of two parents for the simplex crossover operator with $\epsilon = 1\ 5$ and $\epsilon = 2\ 0$ respectively

Figure 3 11 and Figure 3 12 show the evolution of the variance according to the formula (3 72) in

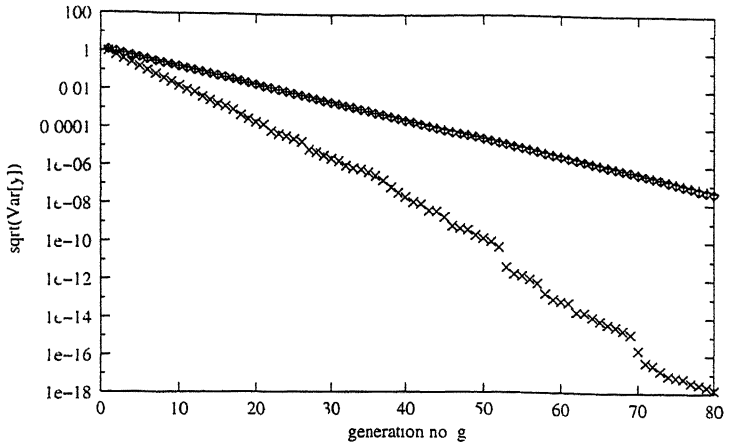


Figure 3 3 Evolution of the square root of the expected population variance for the BLX operator in the flat fitness landscape BLX-0 0 for population size 60 (upper line displayed by +, simulations displayed with a ◇) and 4 (lower dotted line, simulations displayed with a ×) The simulation started with random initialization with variance 1 The results have been obtained by averaging over 2000 (popsize 4) and over 100 (popsize 60) independent runs, respectively

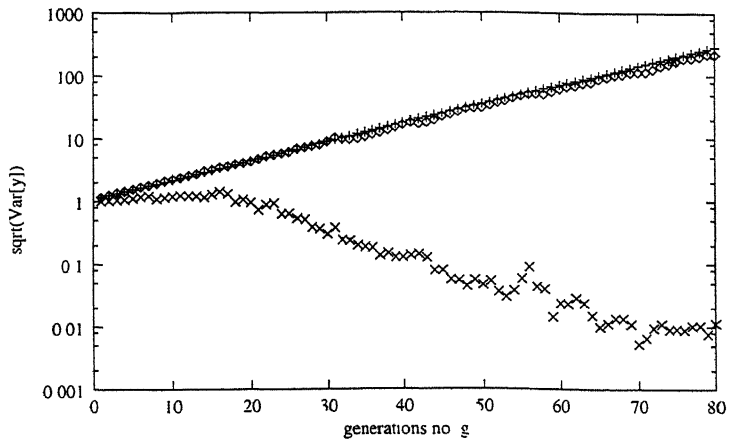


Figure 3 4 Evolution of the square root of the expected population variance for the BLX operator in the flat fitness landscape BLX-0 5 for population size 60 (upper line displayed by +, simulations displayed with a ◇) and 4 (lower dotted line, simulations displayed with a ×) The simulation started with random initialization with variance 1 The results have been obtained by averaging over 2000 (popsize 4) and over 100 (popsize 60) independent runs, respectively

case of three parental recombination of the Simplex Crossover operator with $\epsilon = 2\,1$ and $\epsilon = 2\,25$ respectively

Figure 3 13 and Figure 3 14 compare the variance dynamics of the SPX with five parents according to the equation (3 72) with expansion rate $\epsilon = 2\,5$ and $\epsilon = 2\,75$ respectively with the RGA runs

As is clear from the simulation results, each of these operators is able to exhibit exponentially

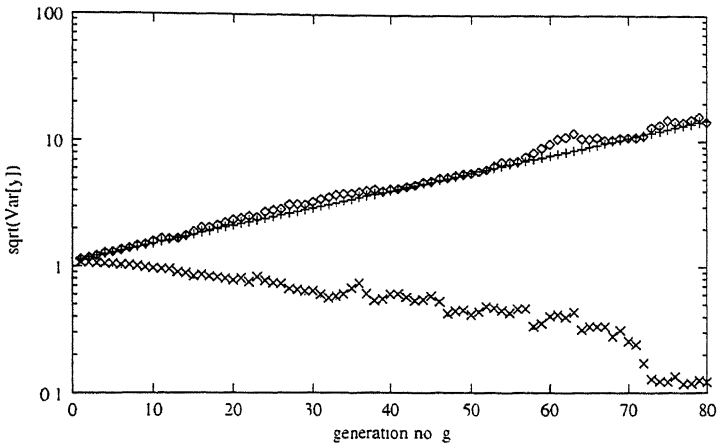


Figure 3 5 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 5$ for population size 60 (upper line displayed with a + simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs

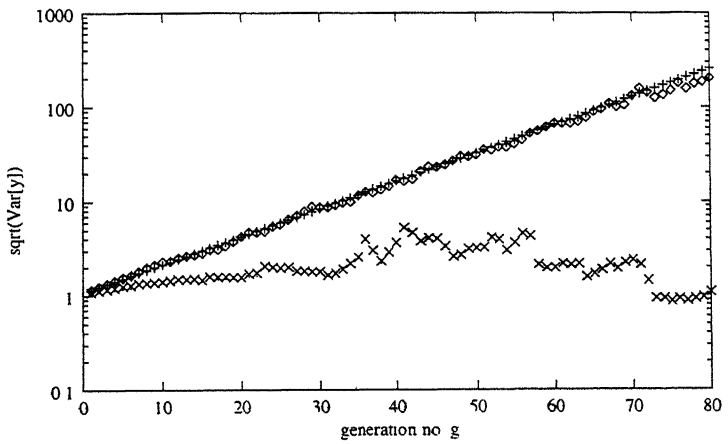


Figure 3 6 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 7$ for population size 60 (upper line displayed with a + simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs

increasing as well as decreasing population variance, depending on the choice of corresponding characteristic parameter value($\eta\ \alpha$,d,ξ_σ,ϵ) The deviation of the simulation from the theoretical predictions are due to accumulation of the statistical errors

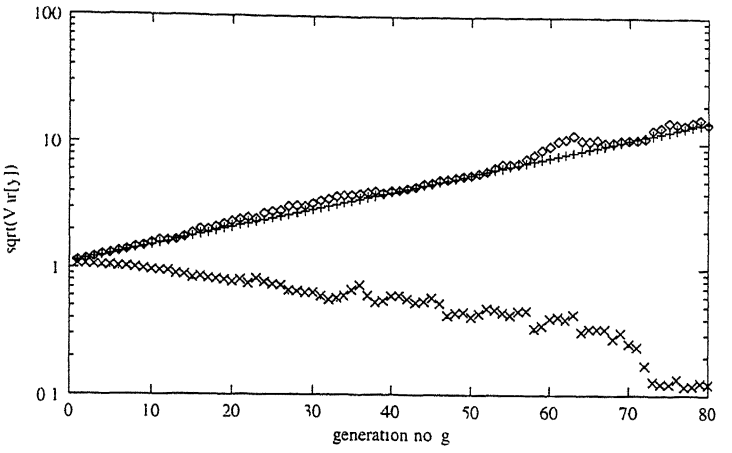


Figure 3 5 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 5$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs

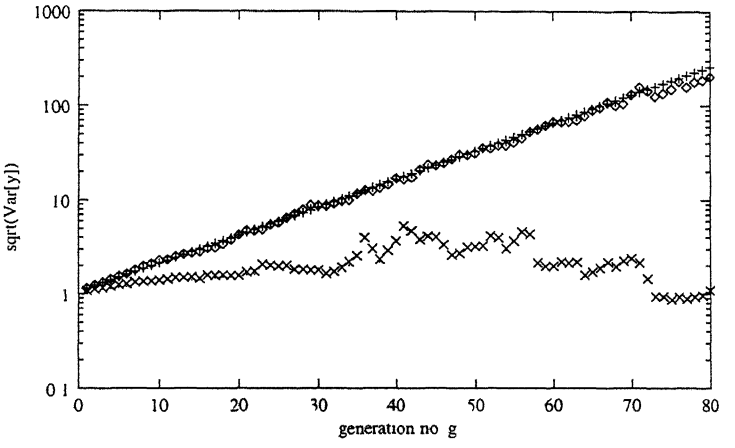


Figure 3 6 Evolution of the square root of the expected population variance for the FR operator in the flat fitness landscape with $d = 0\ 7$ for population size 60 (upper line displayed with a +, simulation shown by a $\diamond$) and for population size 8 (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=8) and 100 (popsize=60) independent runs

increasing as well as decreasing population variance, depending on the choice of corresponding characteristic parameter value(η,α,d,ξ_σ ϵ) The deviation of the simulation from the theoretical predictions are due to accumulation of the statistical errors

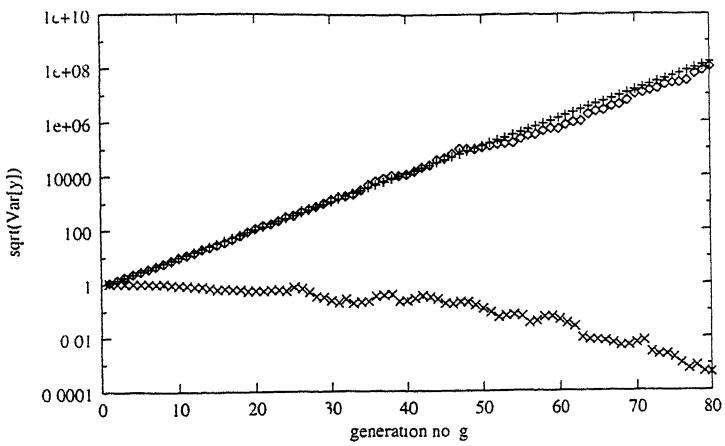


Figure 3 7 Evolution of the square root of the expected population variance for the UNDX crossover operator in the flat fitness landscape with $\sigma_\xi = 0\ 75$ for population size 60 (upper line displayed with a + simulation shown by a $\diamond$) and for population size 6 with $\sigma_\xi = 0\ 55$ (lower dotted line simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=6) and 200 (popsize=60) independent runs

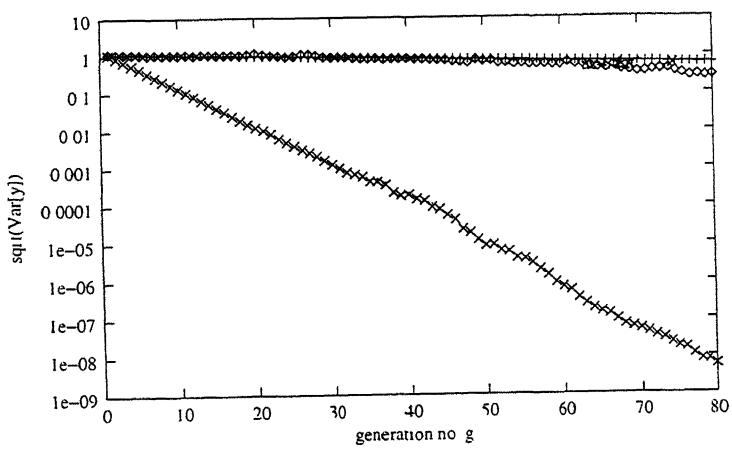


Figure 3 8 Evolution of the square root of the expected population variance for UNDX operator in the flat fitness landscape with $\sigma_\xi = 0\ 50$ for population size 60 (upper line displayed with a + simulation shown by a $\diamond$) and for population size 10 with $\sigma_\xi = 0\ 35$ (lower dotted line, simulation displayed with a $\times$) The simulations started with a random initialization of the population with variance 1 Results have been obtained by running the simulations over 3000 (popsize=10) and 200 (popsize=60) independent runs

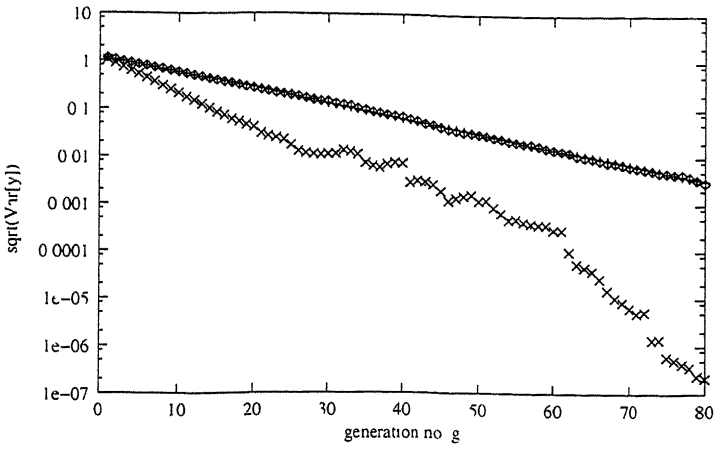


Figure 3 9 Evolution of the square root of the expected population variance for the SPX operator with two parents with $\epsilon = 1\ 5$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=6 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=6) and 300 (popsize=60) independent runs

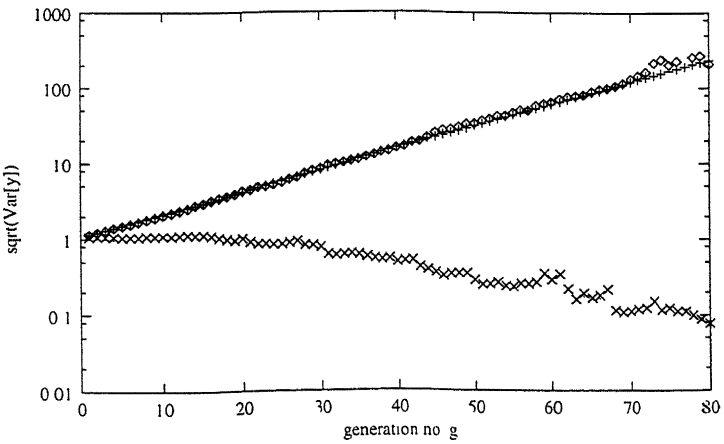


Figure 3 10 Evolution of the square root of the expected population variance for the SPX operator with two parents with $\epsilon = 2\ 0$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=6 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=6) and 300 (popsize=60) independent runs

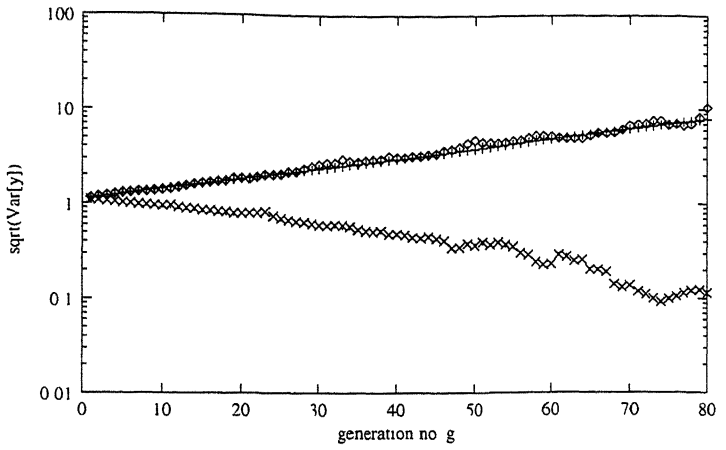


Figure 3 11 Evolution of the square root of the expected population variance for the SPX operator with three parents with $\epsilon = 2\ 1$ for population size=60 (upper line shown with + simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=10) and 200 (popsize=60) independent runs

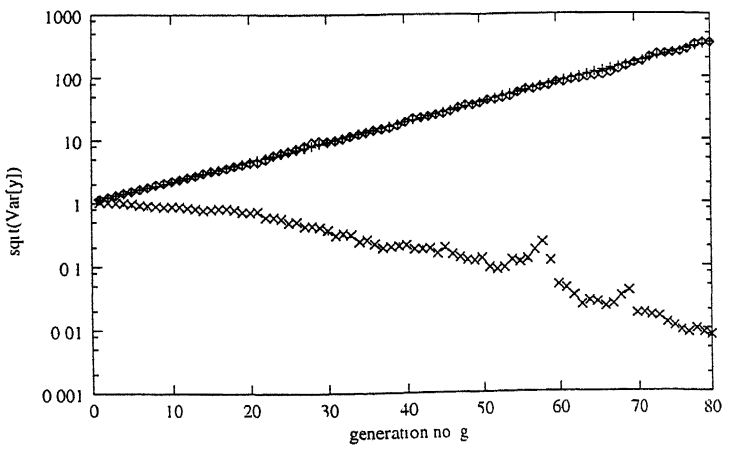


Figure 3 12 Evolution of the square root of the expected population variance for the SPX operator with three parents with $\epsilon = 2\ 25$ for population size=60 (upper line shown with + simulation displayed with a $\diamond$) and population size=5 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs

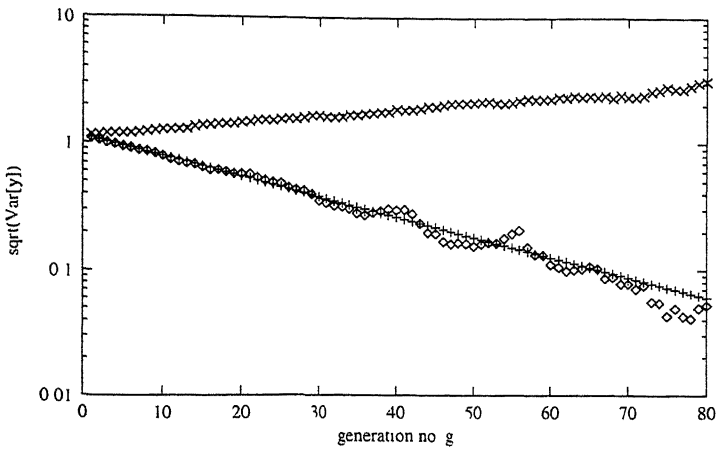


Figure 3 13 Evolution of the square root of the expected population variance for the SPX operator with five parents with $\epsilon = 2\ 5$ for population size=120 (upper line shown with + simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs

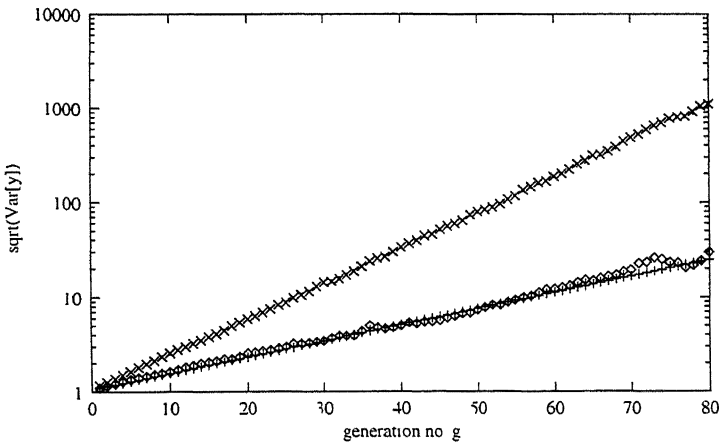


Figure 3 14 Evolution of the square root of the expected population variance for the SPX operator with five parents with $\epsilon = 2\ 75$ for population size=60 (upper line shown with +, simulation displayed with a $\diamond$) and population size=10 (lower dotted line, simulations displayed with a $\times$) The GA runs started with a random initialization of the population with variance 1 The results have been obtained by averaging over 3000 (popsize=5) and 200 (popsize=60) independent runs

Based on the variance evolution equations obtained for different crossover operators and the simulation results it is quite intuitive that above operators will show the similar performance for a particular parameter setting associated with different operators The following table shows the characteristic parameter value of different crossover operators at which these operators would exhibit the identical variance growth in flat fitness landscape The values of the different parameters have been obtained by equating the right side of equations (3 21), (3 24), (3 28), (3 45) and (3 72)

SBX	BLX	FR	UNDX	SPX, M=2		SPX, M=3	
η	α	d	σ_ξ	ϵ, μ small	ϵ, μ very large	ϵ (for small μ)	ϵ (for very large μ)
2 000	0 662	1 095	0 671	$\sqrt{\frac{5\ 400+\frac{3}{\mu}}{1-\frac{1}{\mu}}}$	2 323	$\sqrt{\frac{6\ 400-\frac{4}{\mu}}{1-\frac{1}{\mu}}}$	2 530
3 000	0 500	0 707	0 577	$\sqrt{\frac{4\ 000-\frac{3}{\mu}}{1-\frac{1}{\mu}}}$	2 000	$\sqrt{\frac{5\ 000-\frac{4}{\mu}}{1-\frac{1}{\mu}}}$	2 236
5 000	0 419	0 433	0 530	$\sqrt{\frac{3\ 375-\frac{3}{\mu}}{1-\frac{1}{\mu}}}$	1 837	$\sqrt{\frac{4\ 375-\frac{4}{\mu}}{1-\frac{1}{\mu}}}$	2 092
10 000	0 381	0 226	0 508	$\sqrt{\frac{3\ 103-\frac{3}{\mu}}{1-\frac{1}{\mu}}}$	1 761	$\sqrt{\frac{4\ 103-\frac{4}{\mu}}{1-\frac{1}{\mu}}}$	2 025

Table 3 1 Characteristic parameter values for different crossover operators for similar bahaviour in flat fitness landscape

Here M in case of simplex crossover is the number of parents taking part in crossover Values of the parameter ϵ are presented in two parental and three parental recombination however general expressions for the ϵ can also be obtained from the equation (3 72)

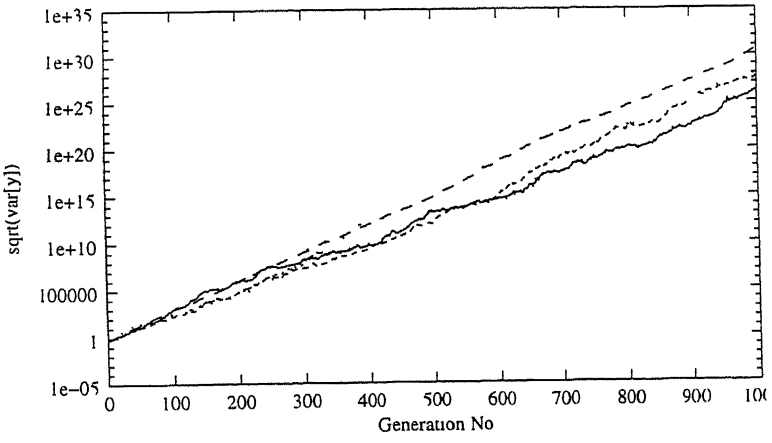


Figure 3 15 The performance of various RCGAs implementing different operators in flat fitness landscape The dotted line is the theoretical line From top the operators are SPX, BLX, SBX, FR and UNDX

Figure 3 15 shows the performance of various RCGAs using different operators in flat fitness landscape The simulation has been performed on a 30-variable problem without any selection and mutation The figure clearly shows that all the operators are performing in a similar manner The deviations from

the theoretical line are due to accumulation of the statistical errors For the simulations the values of characteristic parameters associated with different operators is taken from the table above The values corresponding to the $\eta = 3\ 0$ of SBX operator are taken

3.3 Classification of Crossover Operators

Based on the mean preservation and variance evolution properties the different crossover operators may be classified as below,

Crossover Operator	Mean	Variance
Flat Crossover	Preserves	Always decreases
Simple Crossover	Preserves	Always decreases
Discrete Crossover	Preserves	Always decreases
Arithmetical Crossover	Preserves	Increases for $\mu > \frac{1}{2\lambda(\lambda-1)}$
Linear Crossover	May Preserve	May Increase
BLX-α	Preserves	Increases for $\mu > \frac{3}{2\alpha+2\alpha^2-1}$
Extended Line Crossover	Preserves	Always decreases
Extended Intermediate Crossover	Preserves	Always decreases
Unfair Average Crossover	Preserves	Increases for $\alpha > 1\ 0$
Fuzzy Recombination	Preserves	Increases for $\mu > \frac{3}{d^2}$
Simulated Binary Crossover	Preserves	Increases for $\mu > \frac{1}{2}(\eta+3)(\eta-1)$
Unimodal Normal Distribution Crossover	Preserves	Increases for $\mu > \frac{2}{4\sigma_\xi^2-1}$
Simplex Crossover	Does not preserve	Increases for $\mu > \frac{1+\frac{M-1}{M+1}\epsilon^2}{1+\frac{M-1}{M+1}\epsilon^2-M}$
Linear BGA Crossover	Does not preserve	May Increase

Table 3 2 Crossover Operators for Real-coded Genetic Algorithms

3.3 1 Discussion

In the above table mean preservation means that the mean of the parental population is kept same by the children population and the increase in variance means the children population has increased the variance of the parental population The crossover operator classification above is for finite population size μ In case of Simplex crossover operator, it preserves the mean for infinite population as shown in earlier section The classification has been done according to the analysis performed on the mean and variance evolution of the crossover operators in the earlier sections M in case of Simplex crossover shows the number of parents taking part in crossover Here λ, α ,d, η ,σ_ξ and ϵ represent the different characteristic parameters associated with different operators As for as Fuzzy Connective Based (FCB) operators are concerned the mean preservance and the variance evolution depends on the choice of the fuzzy connectives used for crossover operation As discussed in Chapter 2 that these operators use some fuzzy connective for the creation of new points If each one is treated separately none of the FCB

t-norm	t-conorm	Averaging Function
$T_1(x,y) = min(x,y)$	$G_1(x,y) = max(x,y)$	$P(x,y) = (1-\lambda)x + \lambda y$

Table 3.3 Logical fuzzy connective used in FCB operators

operators will preserve the mean but with certain combination of these connective these can preserve the mean. For example the Logical fuzzy connective are shown in Table 3.3

From the above connectives if we choose T_1 and G_1 to create two children from two parents, the mean would remain the same as that of the parental population and the variance will either decrease or remain same. If P_1 is used to create two children, one with λ and other with $1-\lambda$ the mean will be preserved but the variance in this case will keep on decreasing.

The simplex crossover operator and the linear BGA crossover operator do not preserve the mean for the finite population. The simplex operator however preserves the mean for infinite population.

For UNDX operator only the parallel component along the line joining the parents has been considered in analysis. Though the perpendicular component is not included in the analysis, the relationship between the characteristic parameter of the UNDX operator σ_ξ and population size gives the rough idea of how much value of σ_ξ should be used for variance to increase.

The other things about different operators are as follows,

1. For Arithmetical crossover operator any value of the characteristic parameter $\lambda \leq 1.0$ will decrease the variance independent of the population size μ. For $\lambda > 1.0$ the variance will increase as given by equation (1.36).

2. For Unimodal Normal Distribution Crossover (UNDX) the value of characteristic parameter σ_ξ should be greater than 0.25 to get increasing population variance and for any value $\sigma_\xi > 0.25$ the minimum population size μ is given by equation (1.48). If the $\sigma_\xi \leq 0.25$ the variance of the population will keep on decreasing independent of the population size μ. Note that these values are obtained considering only the parallel component of the crossover operator.

3. The Simplex operator (SPX) does not preserve mean in case of finite population size μ but for infinite population size this operator preserves the mean.

The above analysis has been done considering the *flat fitness landscape*. The *flat fitness landscape* can be regarded as the *local approximations* of the real fitness function to be optimized. Therefore, the analysis describes the local EA behaviour with the general objective to increase the local performance

Chapter 4

Extensions to Other Landscapes

In this Chapter, various real-coded genetic algorithms implementing different crossover operators are tested on *sphere* model and *ridge* model for their similar performances by taking the parameter values from Table 3 1 For the experiments, the second set of characteristic parameters associated with different crossover operators are chosen from the Table 3 1 The different parameters used for the RCGA runs are discussed below,

A population size of 100 is taken for both problems The results have been obtained by averaging over 10 independent runs Number of variables taken is 30 In all the simulations mutation is not used The crossover is done with a probability 1 0 A binary tournament selection is used with tournament size $s = 2$ In case of SBX, BLX and FR the crossover is done variable by variable while in case of UNDX and SPX, line crossover is used The characteristic parameter setting used is as follows,

$$\begin{array}{llll} SBX & \eta & = 3\ 0 \\ BLX & \alpha & = 0\ 5 \\ FR & d & = 0\ 707 \\ UNDX & \sigma_\xi & = 0\ 5773 \\ SPX & \epsilon & = 6\ 13 \end{array}$$

The value of ϵ for SPX operator is determined by taking number of parents $M = 31$

In the following sections the *sphere* and *ridge* problems have been described and the results have been shown

4.1 Sphere Model

The sphere model is most commonly used test function chosen for studying the different algorithms The function used here for simulations is following 30-dimensional function

$$Minimize \ \sum_{i=1}^{30} x_i^2 \tag{4 1}$$

Here the optimal point is at $(0, 0, \quad)$ In the sphere model the problem is of *convergence* All the variables are initialized in $x_i \in (-1\ 0, 1\ 0)$ Real-parameter GAs with SBX, BLX, FR, UNDX and SPX

have been used to optimize this function The Euclidean distance

$$R = \sqrt{\sum_{i=1}^{30} x_i^2}$$

of the best solution in a population from the minimum is determined and is plotted with generation number as shown in Figure 4 1

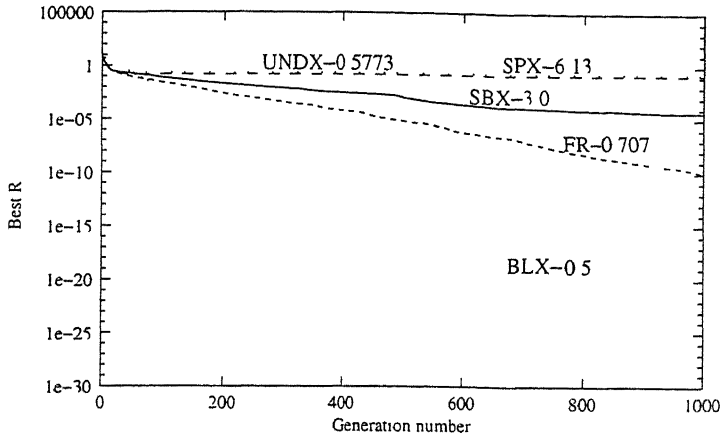


Figure 4 1 Performance of several real-coded GAs with different crossover operators on sphere function

The ordinate axis in the figure above is drawn on the logarithmic scale From the figure it is clear that the RCGAs using SBX, BLX and FR are able to maintain the increased precision with increasing generation and they perform more or less in the same manner Moreover they so a linear convergence order Linear convergence order means an exponential decrease in the residual distance to the minima As for as UNDX and SPX operators are concerned, it is clear from the figure that they get stuck to a particular function value after certain generations They do not perform same as the other operators Note that the values of the parameters associated with different operators obtained are for the flat fitness landscape In flat fitness landscape there is no selection pressure In case of sphere model the role of the selection operator also comes in to picture The UNDX and SPX operators are not performing in the same way with the parameter values as obtained for the flat fitness landscape but these operators can get close to the other operators if the values for their associated parameters is readjusted to some other value Since in case of sphere model there is a problem of convergence so it is intuitive that if we reduce the associated parameter value, the operator should give the enhanced performance

Figure 4 2 shows improved performance of the UNDX operator on sphere model as the value of σ_ξ is reduced from 0 5773 to 0 2 Same improvement in the performance of SPX operator can be seen (Figure 4 3) after ϵ is reduced from 6 13 to 4 0 This is because in case of flat fitness landscape, there is selection pressure and variance of the population is only controlled by the crossover operator so the algorithms using these operators keep on going in one direction i e tthese either increase or decrease the

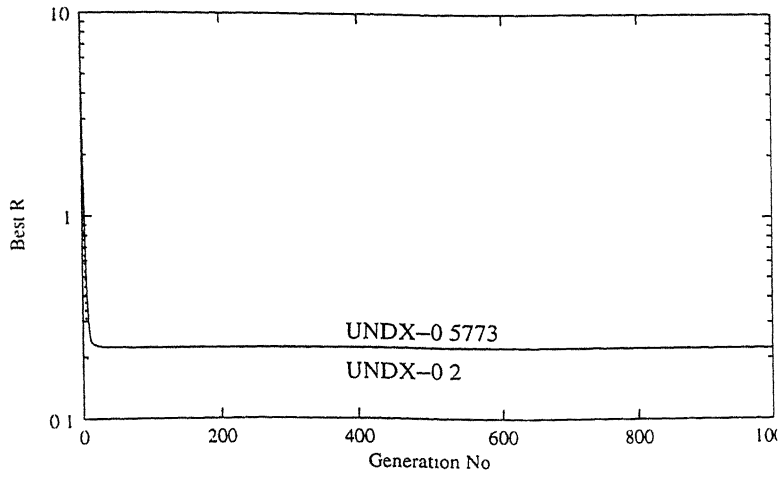


Figure 4 2 Performance of UNDX operator on sphere model after the σ_ξ value is reduced

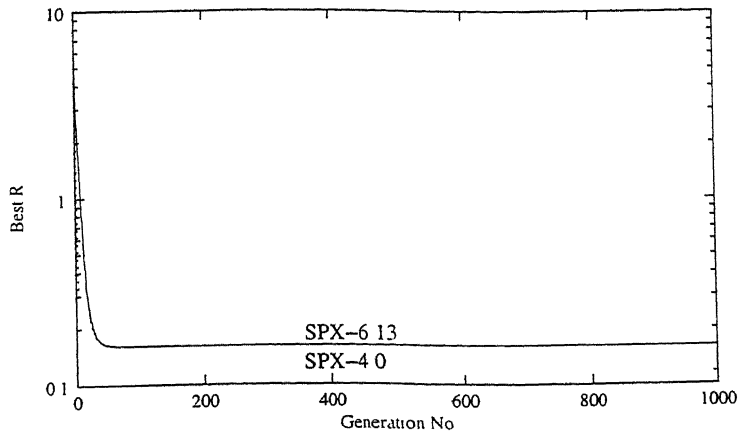


Figure 4 3 Performance of SPX operator on sphere model after the ϵ value is reduced

variance of the population while in case of other fitness landscapes the variance growth of the population is controlled by crossover operator as well as by selection operator In this particular case since the problem is that of minimization, the search regions available to the crossover operator keep on shrinking and therefore the GAs implementing UNDX and SPX operators get stuck So the values obtained for the flat fitness landscape may not give the same performance on other landscapes but the values obtained for the flat fitness landscape give us an idea that in other landscapes the algorithms implementing different crossover operators will perform similarly for some other setting of the characteristic parameters

In the simulations shown above the UNDX and SPX do the crossover on the line formed by the parents while the other operators i e SBX, BLX and FR do crossover variable by variable The variable by variable crossover have also been tested for these crossover operators The results are shown in Figure 4 4 With the variable by variable crossover, it can be seen from the figure that the performance

of GA using UNDX and SPX operators is much improved

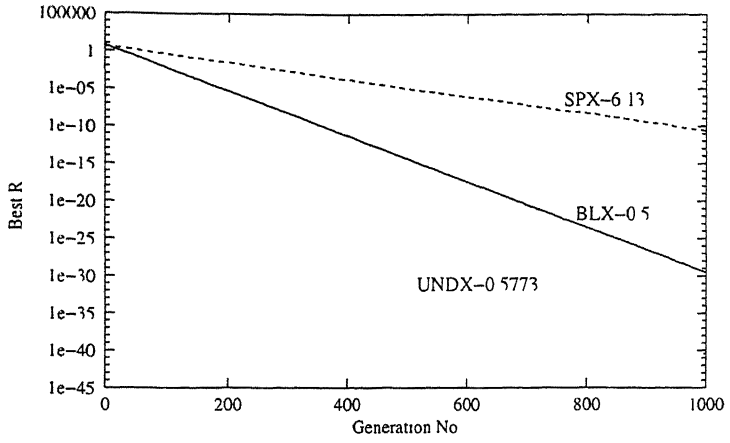


Figure 4 4 The performance of UNDX and SPX operators on sphere model with variable by variable crossover

4.2 Ridge Model

In sphere model, there was a problem of convergence The ridge model is opposite in characteristic The ridge function is defined as,

$$Maximize \quad \mathbf{v}^T\mathbf{x} - d(||(\mathbf{v}^T\mathbf{x})\mathbf{v} - \mathbf{x}||)^\alpha \tag{4 2}$$

where $\mathbf{x}$ is the n-dimensional variable vector and $\mathbf{v}$ is the ridge axis (or the direction vector specifying the ridge axis) Thus, the first term is the projection of $\mathbf{x}$ vector along the ridge axis Since the objective

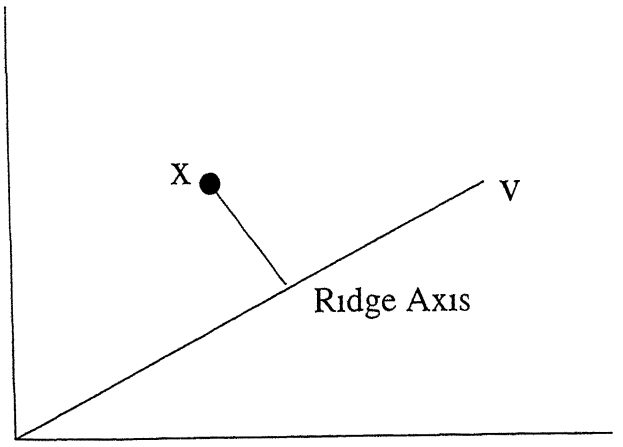


Figure 4 5 Ridge function

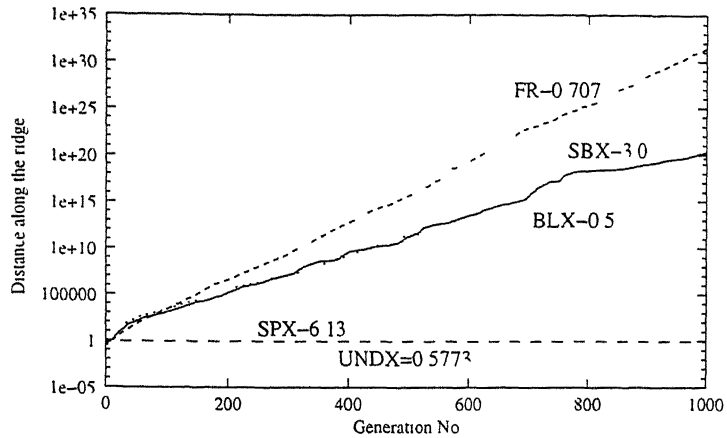


Figure 4 6 The performance of various RGAs with different operators on parabolic ridge function

is to maximize the overall function, the subgoals are to maximize the distance along the ridge axis and minimize the distance normal to the ridge axis from the point (Figure 4 5) The parameters d and α govern the shape of the ridge functions Higher values of d makes the second term more prominent compared to the first term and therefore makes an algorithm difficult to progress along the ridge axis Here a large value of $d = 1$ is used The parameter α has a direct effect on the fitness landscape Usually two values of α are commonly used $\alpha = 2$ is known as the parabolic ridge function and $\alpha = 1$ is known as the sharp ridge function Here only the parabolic ridge is considered

The ridge function has their theoretical maximum solution at infinity along the ridge axis in the search space Since the maximum solution lies on the ridge axis this function tests two aspects

- Converging ability of the algorithm on the ridge axis
- Diverging ability of the algorithm along the ridge axis

In the simulation, the 30-dimensional parabolic ridge function is taken with $d = 1$ The ridge axis a 30-dimensional vector having first element 1 and rest 0 i e $\mathbf{v} = (1, 0 \quad , 0)$ This makes the first coordiante axis the ridge axis The RCGAs with SBX, BLX FR UNDX and SPX are being run on this function The results are shown in Figure 4 6

The best performance is shown by FR operator Here it is clear from figure above that the RCGAs with SBX, BLX and FR show the almost the same performace In the figure the ordiante axis is on the logarithmic scale These algorithms show the linear divergence lines which means an exponential increase in the function value of the function however these are not exactly the same For UNDX and SPX operator, it can be seen that the *RCGAs* implementing these operators also increases the function value but the increament is very small Here again it is found that with the values obtained for flat

fitness landscape the RCGAs are not performing similar to other algorithms This is beacause in case of flat fitness landscape, there is no preference for any area of the search space But here selection operator also comes into work and limits the search to a few regions with more fitness
Although the values obtained for flat fitness landscape is not showing the same behaviour for all the algorithms it is intuitive that in this case some other values of the characterstic parameters associated with the crossover operators may give the similar performance Figure 4 7 shows the performance of SBX and BLX operators on ridge model and values of the associated characterstic parameters with with they show the progress same as that of the FR operator

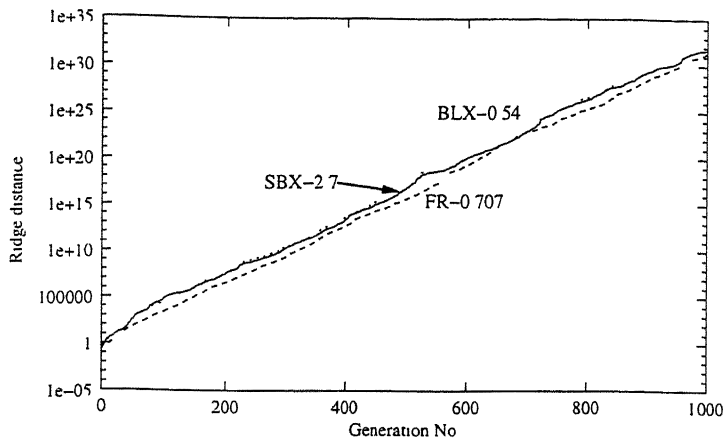


Figure 4 7 RCGAs using SBX and BLX operators performing similar to FR operator on ridge model after altering the associated parameter value

The RCGAs with UNDX and SPX have also been tried to show same performance as that of the FR operator The results are shown in Figure 4 8 for UNDX and in Figure 4 9 for SPX

It is evident from the figure that both of these operators perform in the same way as that of the FR operator after increasing the values of associated characterstic parameter values In case of UNDX operator the value of σ_ξ is increased from 0 5773 to 1 5 In case of SPX operator the value of ϵ is increased from 6 13 to 7 5

4.3 Discussion

In the previous two sections, the performace of different real-coded genetic algorithms have been checked on sphere and ridge model The different RCGAs have been checked whether they will perform in a similar way on different problem landscapes by taking the values of characterstic parameters associated with various crossover operators for which the RCGAs implementing these crossover operators will perform similarly on flat fitness landscape

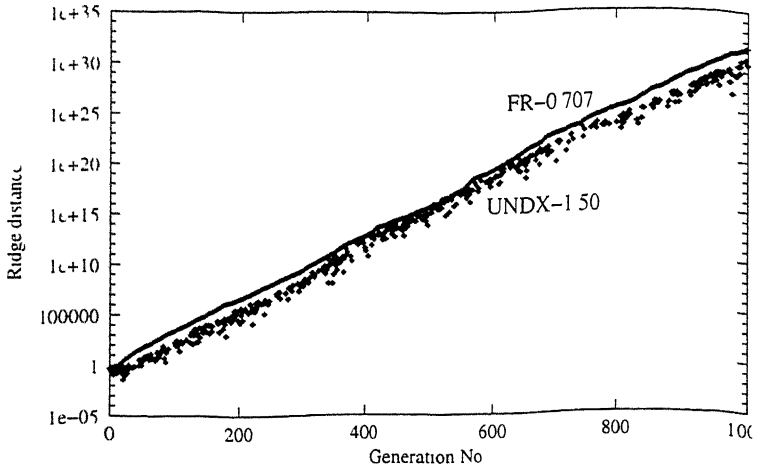


Figure 4 8 Performance of UNDX operator on ridge function after increasing the value of σ_ξ

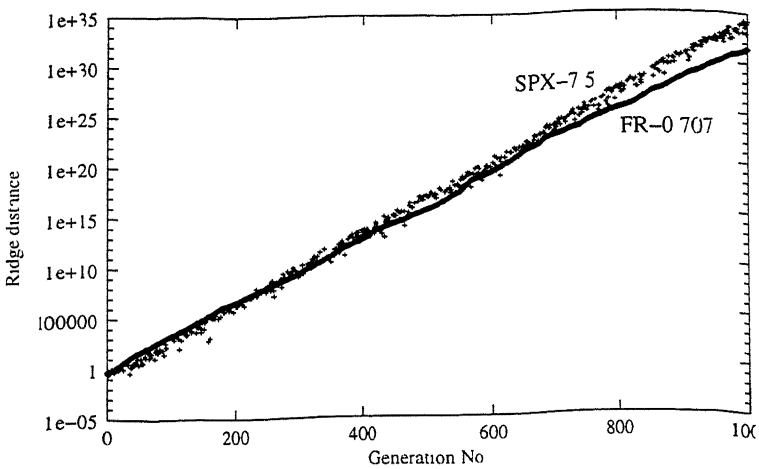


Figure 4 9 Performance of SPX operator on ridge function after increasing the value of ϵ

It has been found that in case of sphere function the best performance is shown by BLX operator with $\alpha = 0\,5$ The SBX and FR operators have also been able to perform in more or less same manner but the UNDX and SPX operators got stuck to a value after certain number generations but the performance of these operators was improved after changing the value of the associated parameter In case of ridge function the best performance is shown by FR operator The RCGAs with SBX and BLX are also performing in more or less the same way In this case too UNDX and SPX are not able to perform in the same way as other operators do Here an increase in the characteristic parameter values of UNDX and SPX operator gives much improved performance and it is almost the same as that of FR operator

This difference can be explained by noting that the analysis of different operators have been done for the flat fitness lanscape In case of flat fitness landscape there is no selective pressure and the crossover operator is only responsible for the increase or the decrease of the population variance therefore all the algorithms are going in the same way In other landscapes such as sphere and ridge model the

selection operator also comes into play and the population variance is controlled by both selection and crossover operators, this is why the algorithms are not able to perform in the same way with the values that have been obtained for the flat fitness landscape But from the simulation results that have been obtained by altering the characteristic parameters values, we are able to get similar performances on other landscapes This shows that the RCGAs with different crossover operators will perform similarly in other landscapes but for some other parameter setting of the associated characterstic parameters Although the values obtained for flat fitness landscape for similar behaviour of RCGAs do not show exactly the same behaviour but these give us an idea which way to go in other landcapes

However a theoretical analysis similar to that in Chapter 3 can also be attempted by considering the selection operator for other landsacapes Unfortunately, such an analysis is dependent on the fitness landscape The important matter resulting from these simulations is that any crossover operator can be used with a proper associated parameter value to match the performance of any other crossover operator Therefore, it may not be necessary to use this or that crossover operator in practice Sticking to one operator and experimenting with it to find out an appropriate parameter value is an ideal practical approach

Chapter 5

Conclusion and Scope for Future Work

The evolutionary algorithms using real number representation for presenting the problem parameters have been found to be very effective in solving the real-world optimization problems The real-coded genetic algorithm uses the real number (floating point) representation and have certain advantages over the traditional binary-coded GA Two main operators in the genetic algorithms are selection and crossover There are a number of crossover operators available in the literature In the present work theoretical analysis of a number of the crossover operators has been performed on flat fitness landscape The crossover operators have been analyzed for their mean preserving and variance evolution characteristics The analysis has been done for flat fitness landscape The analysis has been done on the basis of the postulates proposed by Deb and Beyer (1999) on the *mean* and *variance* characteristic for the self-adaptive evolutionary algorithms in real-valued (unrestricted) search spaces According the postulate on the *mean preservation*, it has been suggested by Beyer and Deb (1999) that with the task of exploration and exploitation, the variance or crossover operator should not change the mean of the parental population The postulate on the *variance* says that the variance of population should exponentially increase by the application of a crossover operator The different crossover operators have been analyzed according to these postulates and the validity of the theoretical predictions has been checked with the simulation runs of the real-GA

5.1 Conclusion

In the present work, crossover operators for *RCGA* namely Simulated Binary Crossover (SBX), Blend Crossover (BLX), Fuzzy Recombination Operator (FR), Unimodal Normal Distribution Crossover (UNDX) and Simplex Crossover (SPX) have been analyzed for the mean and variance characteristics The theoretical predictions have been compared with the simulation results The other operators such as Flat Crossover, Extended Line Crossover, Extended Intermediate Crossover, Arithmetical Crossover have also been analyzed The different crossover operators have been classified as shown in Table 3 2 on the basis of their mean preserving and variance evolution characteristics The values of the characteristic

parameters associated with different operators have been obtained for their similar performance on flat fitness landscape and these have been checked by running simulations on sphere (converging) and ridge (diverging) models

The main observations are as follows,

- All of these crossover operators except the SPX and Linear BGA operator preserve the mean of the parental population in finite population
- The increase or decrease of the population variance by the application of a crossover operator depends on the value of its associated characteristic parameter value and population size
- The operators SBX, BLX, FR, SPX, UNDX and Arithmetical Crossover are able to exponentially increase or decrease the population variance depending on choice of the associated characteristic parameter and the population size
- The operators like Flat Crossover, Extended Line Crossover, Extended Intermediate Crossover Simple Crossover, Discrete Crossover always decrease the population variance no matter how much population size is used
- The crossover operators Arithmetical Crossover and Fuzzy Connective Based (FCB) operators are deterministic in there working i e for any two given parents these operators always produce the same child
- In case of the UNDX operator the characteristic parameter σ_ξ should be greater than 0 5 for the variance of the population to increase Any value less than 0 5 would always decrease the population variance independent of the population size
- For arithmetical operator the value of characteristic parameter λ must be greater than 1 0, any value less or equal to 1 0 will always decrease the population variance
- Equations (3 25), (3 28), (3 32), (3 48), (3 69) give the minimum value of the population size in case of SBX, BLX, FR, UNDX and SPX respectively It can be seen except SBX for the other operators the population size μ is inversely proportional to the associated characteristic parameter so with increasing value of the parameter, the population size will keep on decreasing and may come down to the values of 2 or 3 In such a case if we want to decrease the population variance we can not do this in these operators but in case of SBX the population size increases with the increase in the value of the characteristic parameter η so it easier to control the search in SBX than other operators

- The simulation results show that the operators SBX, BLX, FR and UNDX follow the postulates on mean preservation and variance evolution (Beyer and Deb, 1999) therefore these operators are good candidate for use with RCGAs

- The results obtained for the sphere and ridge models show that the with the values obtained for similar bahaviour in flat fitness landscape, the algorithms implementing different crossover operators do not perform exactly in the same manner

- It has been found that like flat fitness landscape the different RCGAs behave in the same manner i e they show same progress towards the optima but for a different combination of the characteristic parameters associated with the crossover operators

- The values obtained in case of flat fitness landscape give us an idea about what values of the parameters to use in other landscape to get good performance e g if we have some problem to optimize we can know from the equations and the values obtained for flat fitness landscape as what parameter value of the crossover operator to be used corresponding to chosen population size so as to get increasing variance of the population So this value give us the minimum value of the parameter for which the algorithm will provide good performance

- The main point that results from the analysis and simulations in the present work is that all the GAs implementing various crossover operators will perform similarly in different landscapes with some setting of the parameter values associated with the crossover operators This implies that there should not be any preference of a crossover operator over some other because the other operator will give similar performance for some value of the parameter of that operator So sticking to a single operator and experimenting with the different values of the characteristic parameter is an ideal practical approach

This shows that the processing of a *RCGA* can be controlled by controlling the value of the characteristic parameter associated with the crossover operators for the particular population size
The values of the different parameters associated with different crossover operators are listed in Table 3 1 The table shows the parameter values associated with the different crossover operators for which they will perform similarly on flat fitness landscape
With these observations, the following guidelines for designing of an efficient crossover operators are given

1 The crossover operator should preserve the mean of the parental population

2 The crossover operator should be able to exponentially increase the variance of population

The validity of above guidelines can also be explained with the help of the FCB crossover operators The fuzzy connectives used in the FCB crossover operators are shown in the Table 2 1 Among the four families Herrera et al have reported that the *Logical* fuzzy connectives were performing better than others on most of the problems In these operators the children are created by choosing the combinations of the fuzzy connectives If we take t-norm and t-conorm fuzzy connective, the mean of the population would be preserved and the variance will either decrease or remain same If we take the averaging function the two children created by taking values λ and $1-\lambda$ would preserve the mean of the parental population and the variance will be decreased The fuzzy connectives of the other families do not preserve the mean except the t-norm and t-conorm of the *Algebraic* family Thus it can be seen that the mean preservation is a desirable property of the EA As for as the second guideline is concerned Herrera et al tested these operators on the functions for which the location of the optimum solution is already known So a controlled decrease in the population variance will serve the purpose but in real-world optimization problems the information of the location of the optima is usually not known so it would be better if the crossover operator has the ability to increase the population variance The necessary decrease in the population variance will be induced by the selection operator
With respect to the above guidelines the Simulated Binary Crossover Operator (SBX) the Blend Crossover (BLX), the Fuzzy Recombination (FR), Unimodal Normal Distribution Crossover (UNDX) are good candidates for use with real-coded genetic algorithms since these operators fulfill both of the above properties

5.2 Scope for Future Work

On the basis of the present work and results, the following tasks can be accomplished,

- The different evolutionary algorithms employing the different crossover operators could be tested for similar performance for the landscapes other than sphere and ridge taking the values of characteristic parameters from the table 3 1
- The EAs using different crossover operators can be tested on test problems to get a clearer picture of the properties of the crossover operator
- Presently in the real-coded GAs, the characteristic parameter is kept constant for whole of the GA run With the variance evolution equations in hand, new evolutionary algorithms having a varying characteristic parameter depending on the variance of the current and previous populations could be designed and tested

- The concept of the invariance suggested by Hansen (2000) for *evolution strategies* can be used with real-coded GAs having different crossover operators to test their invariance properties

Appendix A

Calculation of Variance for the Crossover operators as in Beyer and Deb (1999)

Considering the k^{th} sum terms in (3 11) Using eqns (3 1),(3 2) and (3 10), we get,

$$y_{1,k} - \langle y \rangle = \left(\frac{1}{2} - \frac{\beta_k}{2}\right) x_{1,k} + \left(\frac{1}{2} + \frac{\beta_k}{2}\right) x_{2,k} - \frac{1}{\mu}\sum_{l=1}^{\mu/2}(x_{1,l} + x_{2,l})$$

or

$$y_{1,k} - \langle y \rangle = \left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right) x_{1,k} + \left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right) x_{2,k} - \frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l}) \quad (5\ 1)$$

and

$$y_{2,k} - \langle y \rangle = \left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right) x_{1,k} + \left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right) x_{2,k} - \frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l}) \quad (5\ 2)$$

Squaring eqns (5 1) and (5 2) gives,

$$(y_{1,k} - \langle y \rangle)^2 = \left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right)^2 x_{1,k}^2 + \left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right)^2 x_{2,k}^2 x_{2,k}^2 + \frac{1}{\mu^2}\left(\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})\right)^2$$

$$+2\left[\left(\frac{1}{2} - \frac{1}{\mu}\right)^2 - \frac{\beta_k^2}{4}\right] - 2\left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right) x_{1,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

$$-2\left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right) x_{2,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

$$(y_{2,k} - \langle y \rangle)^2 = \left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right)^2 x_{1,k}^2 + \left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right)^2 x_{2,k}^2 x_{2,k}^2 + \frac{1}{\mu^2}\left(\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})\right)^2$$

$$+2\left[\left(\frac{1}{2} - \frac{1}{\mu}\right)^2 - \frac{\beta_k^2}{4}\right] - 2\left(\frac{1}{2} - \frac{1}{\mu} + \frac{\beta_k}{2}\right) x_{1,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

$$-2\left(\frac{1}{2} - \frac{1}{\mu} - \frac{\beta_k}{2}\right) x_{2,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

Putting above two eqns together, we get

$$(y_{1,l} - \langle y \rangle)^2 + (y_{2,k} - \langle y \rangle)^2$$

$$= 2\left[\left(\frac{1}{2} - \frac{1}{\mu}\right)^2 + \frac{\beta_k^2}{4}\right](x_{1,k}^2 + x_{2,k}^2) + \frac{2}{\mu^2}\left(\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})\right)^2$$

$$+4\left[\left(\frac{1}{2} - \frac{1}{\mu}\right)^2 - \frac{\beta_k^2}{4}\right]x_{1,k}x_{2,k} - 4\left(\frac{1}{2} - \frac{1}{\mu}\right)x_{1,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

$$-4\left(\frac{1}{2} - \frac{1}{\mu}\right)x_{2,k}\frac{1}{\mu}\sum_{l\neq k}^{\mu/2}(x_{1,l} + x_{2,l})$$

Substituting the above equation in eqn (3 8), we obtain

$$\sigma^2[y] = \frac{1}{\mu}\sum_{k=1}^{\mu/2} E\left[(y_{1,k} - \langle y \rangle)^2 + (y_{2,k} - \langle y \rangle)^2\right]$$

$$= \frac{1}{\mu}\sum_{k=1}^{\mu/2}\left[1 - \frac{2}{\mu} + \overline{\beta_k^2}\right]\left(\overline{x^2} - \overline{x}^2\right)$$

$$= \frac{1}{\mu}\frac{\mu}{2}\left[1 - \frac{2}{\mu} + \overline{\beta_k^2}\right]\left(\overline{x^2} - \overline{x}^2\right)$$

or

$$\sigma^2[y] = \left[\frac{1}{2} - \frac{1}{\mu} + \frac{\overline{\beta_k^2}}{2}\right]\left(\overline{x^2} - \overline{x}^2\right) \tag{5 3}$$

Since β_k are the samples from the same distribution, we can write $\overline{\beta_k^2} = \overline{\beta^2}$ and taking eqn (3 9) into account, we can write the following evolution equation of the expected population variance

$$\sigma^{2(g+1)} = \frac{1}{2}\left(1 - \frac{2}{\mu} + \overline{\beta^2}\right)\sigma^{2(g)} \tag{5 4}$$

Bibliography

[1] Back T (1996) Evolutionary Algorithms in Theory and Practice *Oxford University Press*

[2] Beyer H G and Deb, K (1999) On Self-Adaptive Features in Real-Coded Evolutionary Algorithms

[3] Deb, K and Agrawal, R B (1995) Simulated Binary Crossover for Continuous Search Space *Complex Systems* 9 115-148

[4] Deb K and Goyal M (1996) A Combined Genetic Adaptive Search (GeneAS) for Engineering Design *Transactions of the ASME Journal of Mechanical Design*, 120(2), 162-164

[5] Deb, K and Beyer, H G (1999) ,Self-Adaptive Genetic Algorithms with Simulated Binary Crossover *Evolutionary Computation,* 2000

[6] Eshelman, L J and Schaffer, J (1993) Real-Coded Genetic Algorithms and Interval Schemata *Foundations of Genetic Algorithms 2* edited by D Whitley (Morgan Kaufmann, San Mateo) 187-202

[7] Eshelman, L J , Mathias, K E and Schaffer, J D (1997) Crossover Operator Biases Exploiting the Population Distribution *Proceedings of the Seventh International Conference on Genetic Algorithms* T Back (Ed) (Morgan Kaufman Publishers, San Francisco), 354-361

[8] Goldberg D E (1990) Real-Coded Genetic Algorithms, Virtual Alphabets, and Blocking *Complex Systems* 5 139-167

[9] Hansen, N (2000) Invariance, Self-Adaptation and Correlated Mutations in Evolution Strategies *Proceedings of Parallel Problem Solving from Nature VI Conference, 2000*, 355-364

[10] Herrera, F Lozano M and Verdegay, J L (1997) Fuzzy Connective Based Crossover Operator to Modal Genetic Algorithms Population Diversity *Fuzzy Sets and Systems* 92(1), 21-30

[11] Herrera, F , Lozano M , and Verdegay, J L (1998) Tackling Real-Coded Genetic Algorithms Operators and Tools for Behavioural Analysis *Artificial Intelligence Review* 12, 265-319

[12] Iba H Akiba S , Higuchi, T and Sato, T (1992) BUGS A-Bug Based Search Strategy using Genetic Algorithms *Parallel Problem Solving form Nature 2*, R Manner and B Manderick (Eds) (Elsevier Science Publishers, Amsterdam), 165-174

[13] Janikow C Z and Michalewicz, Z (1991) An Experimental Comparison of Binary and Floating Point Representation in Genetic Algorithms *Proceedings of 4th International Conference on Genetic Algorithms*, R Belew and L B Booker (Eds) (Morgan Kaufmann, San Mateo) 31-36

[14] Nomura T (1997) An Analysis on Crossovers for Real Number Chromosomes in an Infinite Population Size *Proceedings of International Joint Conference on Artificial Intelligence* (IJCAI 97) 936-941

[15] Ono, I and Kobayashi, S (1996) A Real-Coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover *Proceedings of the 7th International Conference on Genetic Algorithms*, 246-253

[16] Ono I , Kita, H and Kobayashi, S (1998) Theoretical Analysis of the Unimodal Normal Distribution Crossover for Real-Coded Genetic Algorithms *IEEE International Conference on Evolutionary Computation*, 529-534

[17] Ono, I , Kita, H and Kobayashi, S (1999) Multi-Parental Extension of the Unimodal Normal Distribution Crossover for Real-Coded Genetic Algorithms *Congress on Evolutionary Computation* 1581-1587

[18] Radcliffe N J (1991a) Equivalence Class Analysis of Genetic Algorithms *Complex Systems* 5(2) 183-205

[19] Radcliffe, N J (1991b) Forma Analysis and Random Respectful Recombination *Proceedings of 4th International Conference on Genetic Algorithms*, R Belew and L B Booker (Eds) (Morgan Kaufmann San Mateo), 222-229

[20] Tsutsui, S , Yamamura, M and Higuchi, T (1999) Multi-Parent Recombination with Simplex Crossover in Real-Coded Genetic Algorithms *Proceedings of the Genetic and Evolutionary Computation Conference*, Volume 1, 657-664

[21] Tsutsui, S , Higuchi, T and Yamamura, M (2000) Theoretical Analysis of Simplex Crossover for Real-Coded Genetic Algorithms *Proceedings of Parallel Problem Solving from Nature VI Conference, 2000* 365-374

[22] Schlierkamp-Voosen D (1994) Strategy Adaptation by Competition *Proceedings of 2nd European Congress on Intelligent Techniques and Soft Computing*, 1270-1274

[23] Storn R and Price, K (1997) Differential Evolution - A simple evolution strategy for fast optimization *Dr Dobb's Journal 22(4), April, 97*, 18-24 and 78

[24] Voigt H M Muhlenbein, H and Cvetkovic, D (1995) Fuzzy Recombination for the Breeder Genetic Algorithm *Proceedings of 6th International Conference on Genetic Algorithms*, Edited by L J Eshelman, (Morgan Kaufman Publishers, San Francisco), 104-111

[25] Wright, A (1991) Genetic Algorithms for Real Parameter Optimization *Foundations of Genetic Algorithms 1*, G J E Rawlin (Ed) (Morgan Kaufmann, San Mateo) 205-218

[26] Yang J M , Lin, C J and Kao, C Y (2000) A Robust Evolutionary Algorithm for Global Optimization *International Journal of Neural Systems, Vol 10, No 5 (2000)* 333-352 (C) World Scientific Publishing Company

A133712

A133712

Date Slip

The book is to be returned on the date last stamped

A133712

TH
ME/2001/M
G959a
A133712